Circuitos Digitais

Engenharia Elétrica/Engenharia de Automação/
Engenharia de Computação/Sistemas de Informação/
Ciência da Computação/Tecnologia de Redes

Prof. VICTOR MARQUES MIRANDA

CONTEÚDOS

I. Conceitos Básicos de Sistemas Digitais

II. Sistemas de Numeração

II.1. Conversões entre Bases

II.2. Operações Aritméticas

III. Portas Lógicas e Formas de Representação de uma Função Lógica

IV. Álgebra Booleana e Simplificação de Circuitos

V. Redes Combinacionais e Minimização Lógica

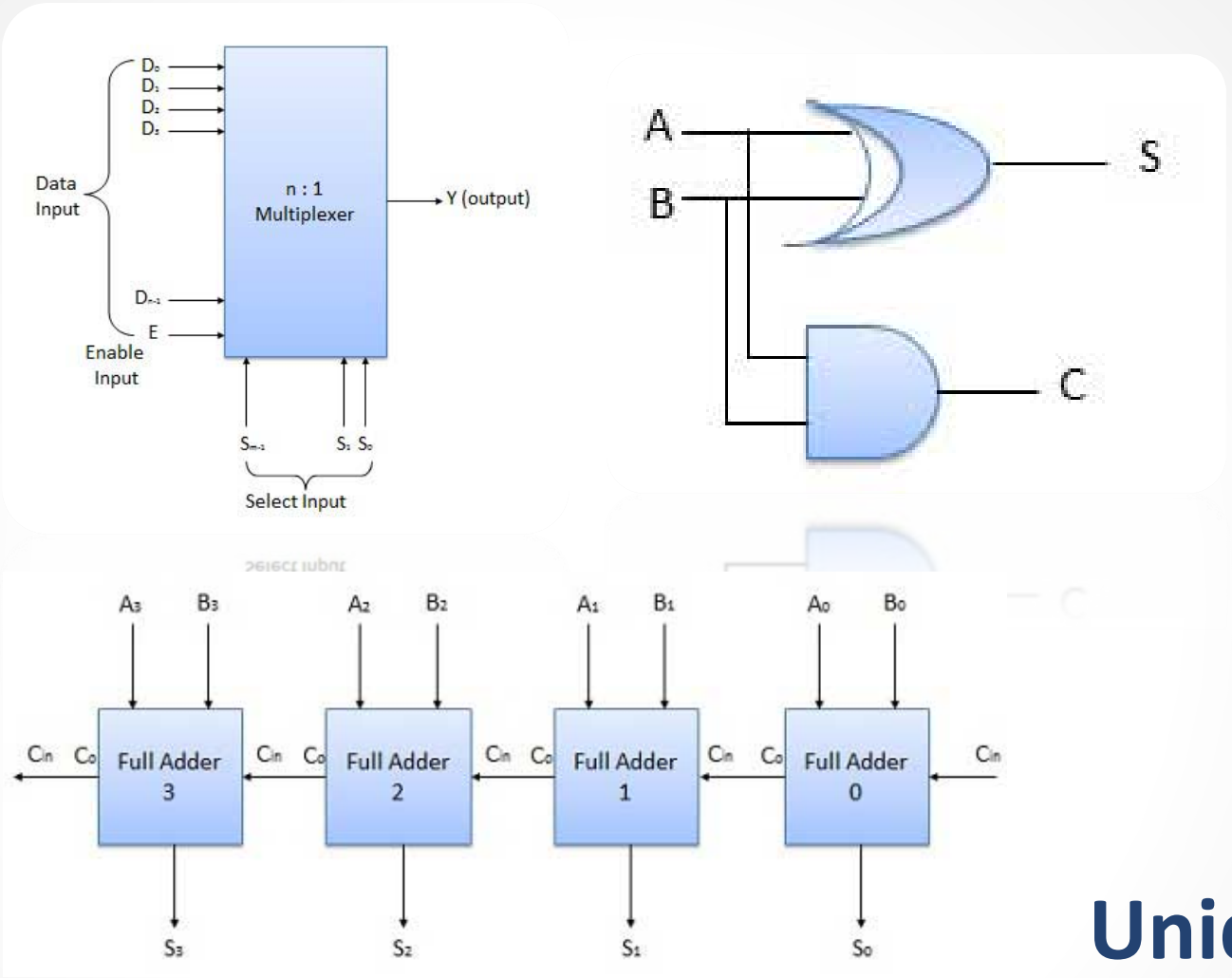
VI. Projeto Lógico Combinacional

VII. Módulos-Padrão Combinacionais e Aritméticos

VIII. Sistemas Sequenciais – Parte 1: Máquinas de Estados, Elementos de Memória e Análise e Projeto de Redes Sequenciais Canônicas

IX. Sistemas Sequenciais – Parte 2: Módulos-Padrão – Contadores

X. Revisão dos Conteúdos e Aplicação da N2



Unidade 7

Módulos-Padrão Combinacionais e Artiméticos

Módulos-Padrão e Aritméticos Combinacionais

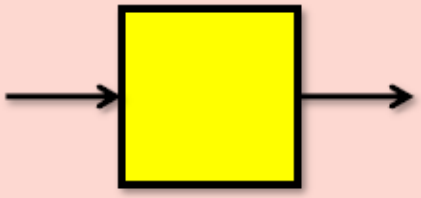
Hardware Digital

- Somadores, Subtratores
- Codificadores, Decodificadores
- Multiplexadores
- Comparadores
- Deslocadores
- Verificadores de Paridade

Circuitos Lógicos

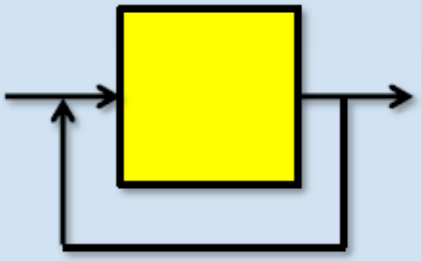
Circuitos combinatórios

suas saídas dependem unicamente de suas entradas



Circuitos sequenciais

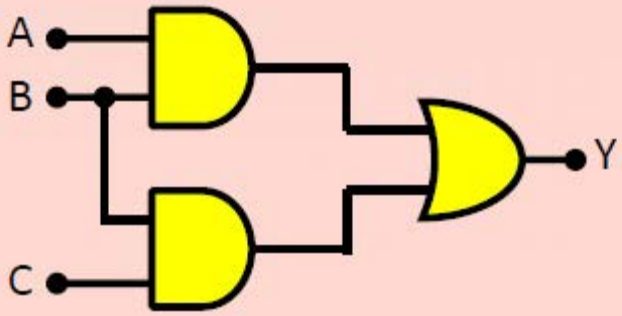
estado lógico atual depende das entradas atuais e do estado lógico anterior



Circuitos Lógicos

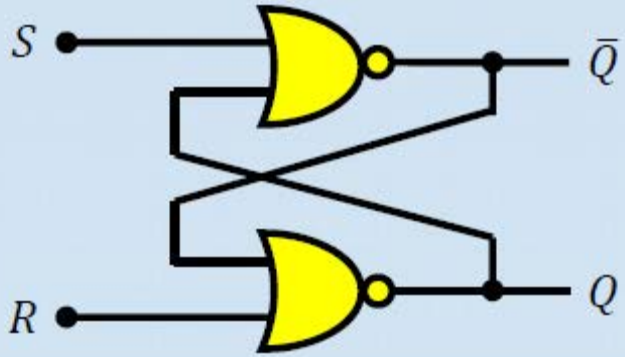
Combinatórios

- A saída de depende apenas da entrada atual
- Não existe temporização, a não ser o retardo de propagação do sinal pelas portas
- Não incluem informação de estado



Sequenciais

- A saída depende não apenas da entrada atual, mas também do estado anterior do circuito
- Incluem informação de estado
- Serão discutidos mais tarde



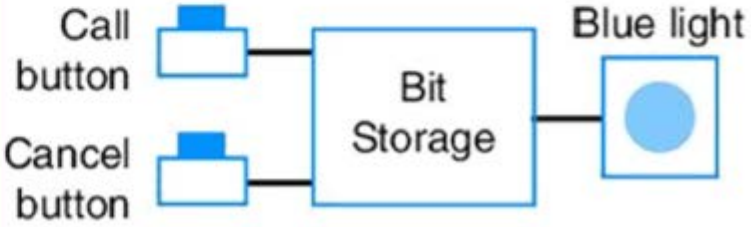
2 Cenários

Campainha de casa

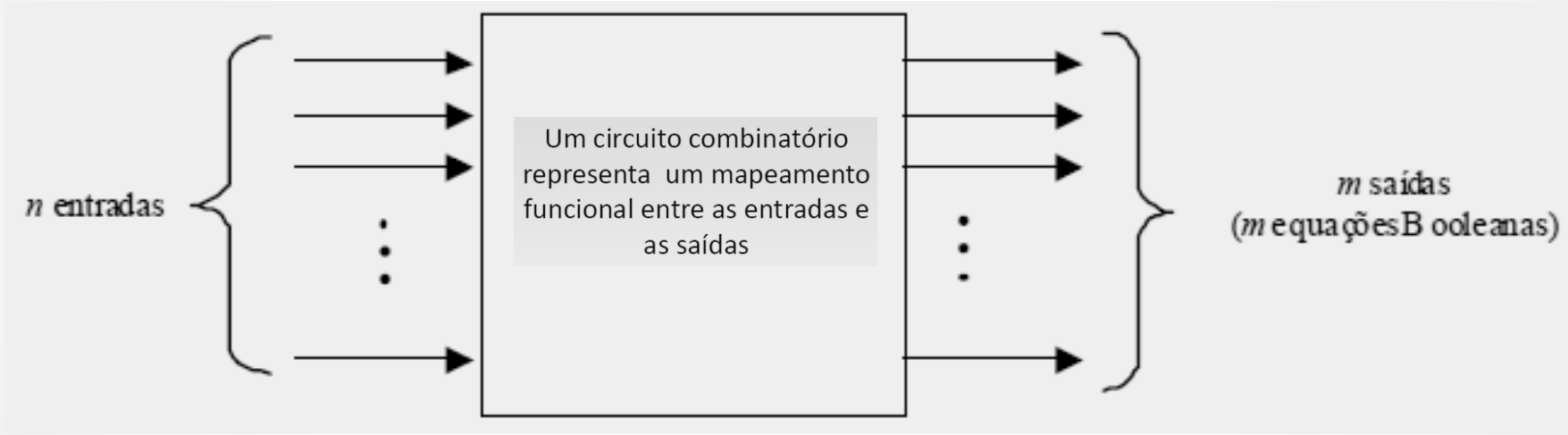
- A campainha tocará sempre que pressionada

Campainha de avião

- Mesmo após o passageiro soltar o botão de chamar, a luz indicadora "lembra" de ficar acesa.



Módulos Padrão Combinacionais



MÓDULOS ARITMÉTICOS:

SOMADORES

Somadores

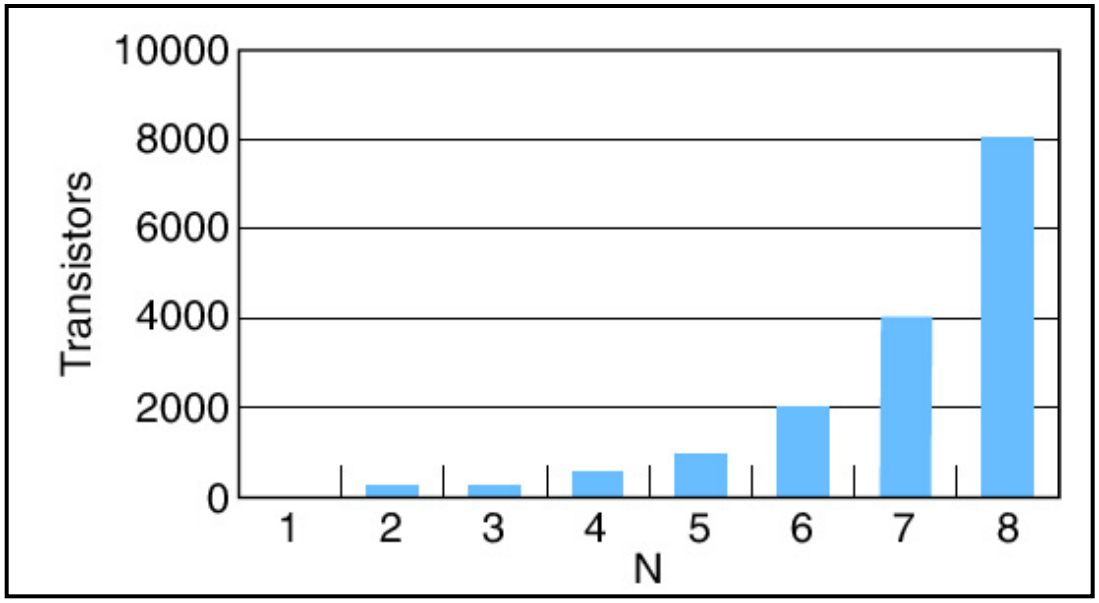
- Um somador de N bits é um componente de bloco operacional que:
  - Adiciona dois vetores de N bits, A e B, gerando uma soma S de N bits e
  - Um transporte (o “vai um”) C de 1 bit

Inputs				Outputs		
a1	a0	b1	b0	c	s1	s0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0

Inputs				Outputs		
a1	a0	b1	b0	c	s1	s0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

Somadores

- Para somadores com um maior número de dígitos, a tabela verdade passa a ser de difícil construção (somador de 16 bits tem mais de 4 bilhões de linhas).
- Por esta razão, somadores não são construídos usando a lógica de 2 níveis.



Meio Somador ou Somador Parcial

Um meio-somador (***Half Adder - HA***) é um componente combinacional que:

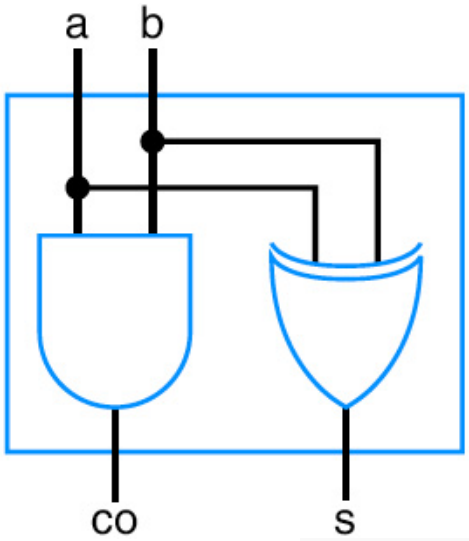
Adiciona dois bits (a e b) e gera uma soma (S) e um bit de transporte de "vai um" (ou *carry out*).

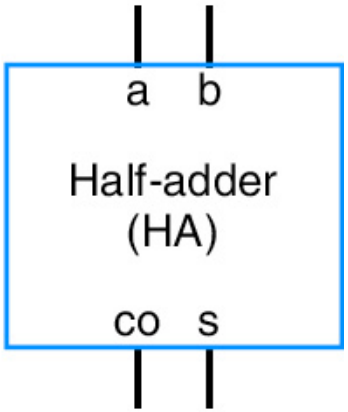
Inputs		Outputs	
a	b	co	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

carry out = co = vai 1 = = ab

soma = s = = a'b+ab'

- Observando a tabela verdade abaixo, note que podemos implementar:
  - O bit carry-out com uma porta AND.
  - O bit de soma com uma porta XOR.

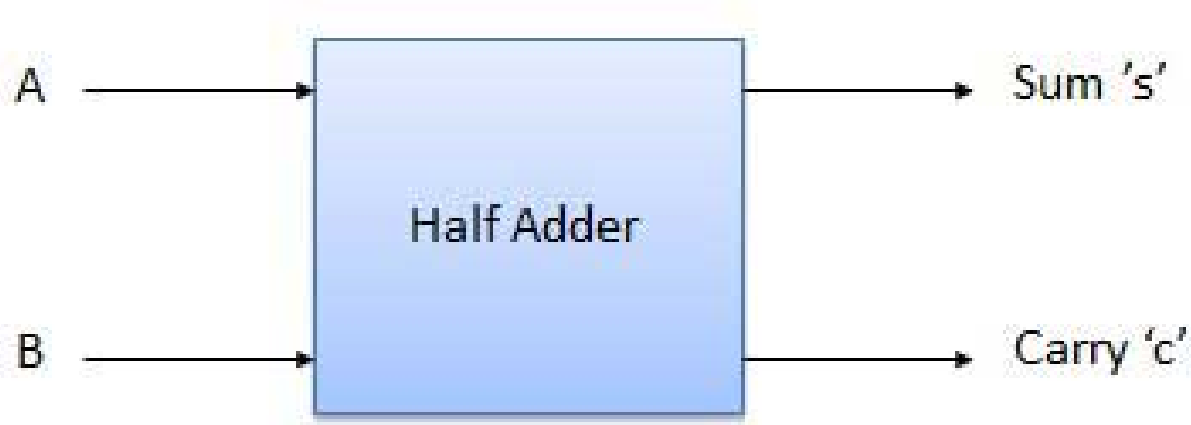




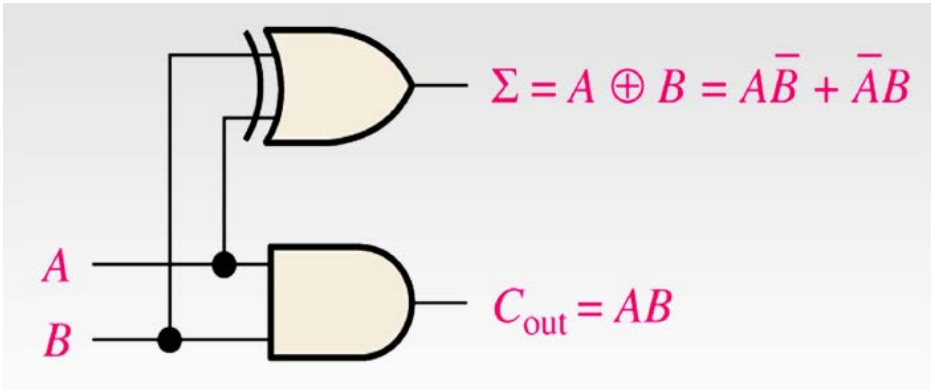
$C_{out} = A \cdot B$

$\Sigma = A \oplus B$

Meio Somador de 1 Bit



Inputs		Output	
A	B	S	C
0	0	0	0
0	1	1	0
0	1	1	0
1	1	0	1



Somador Completo (*Full Adder*)

Modo 1:

- Um somador completo é um componente combinacional que
  - Adiciona três bits (a, b e o *carry-in* ou "vem-1" ou ci) e gera uma soma (s) e
  - Um bit de transporte de "vai um" (ou *carry out*).

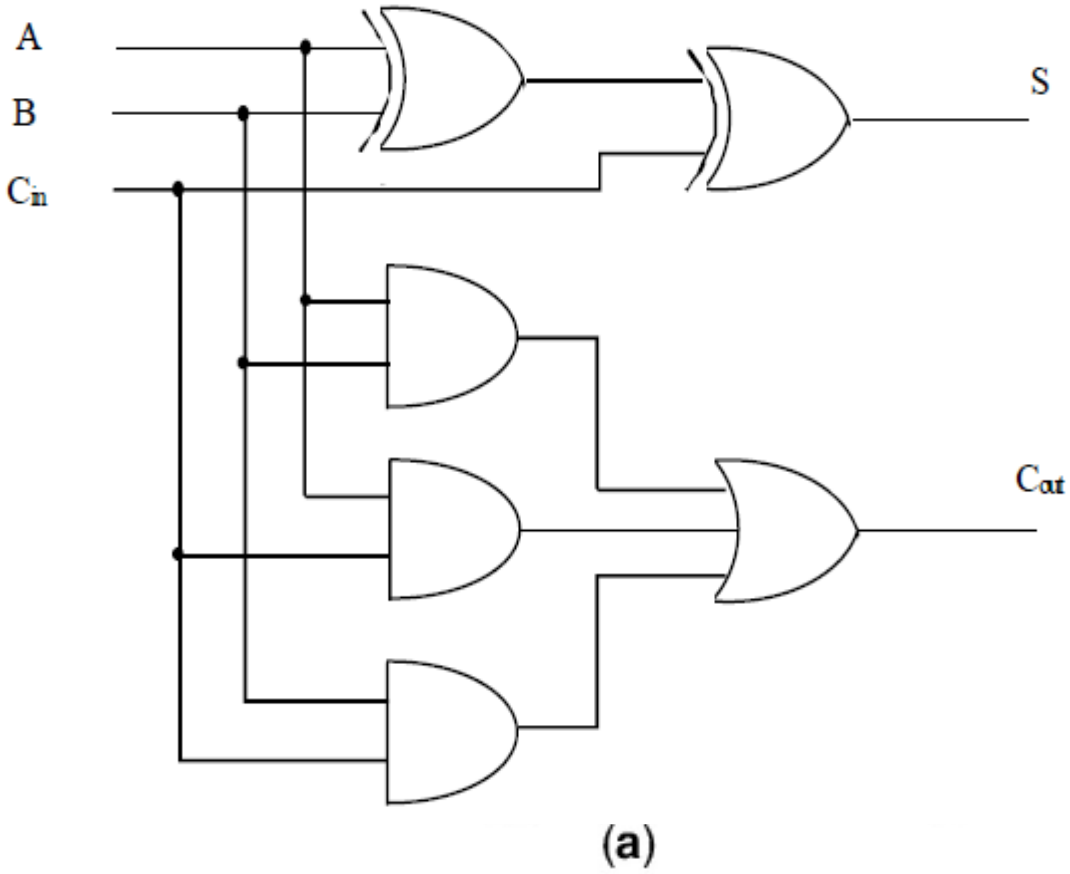
  - co = a'bc + ab'c + abc' + abc
  - co = a'bc + **abc** + ab'c + **abc** + abc' + **abc**
  - co = (a'+a)bc + (b'+b)ac + (c'+c)ab
  - **co = bc + ac + ab**

  - s = a'b'c + a'bc' + ab'c' + abc
  - s = a'(b'c + bc') + a(b'c' + bc)
  - s = a'(b xor c) + a(b xor c)'
  - **s = a xor b xor c**

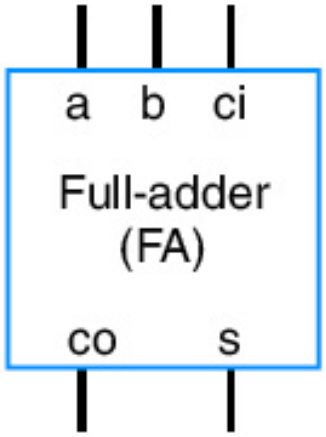
Inputs			Outputs	
a	b	ci	co	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Somador Completo

Modo 1:

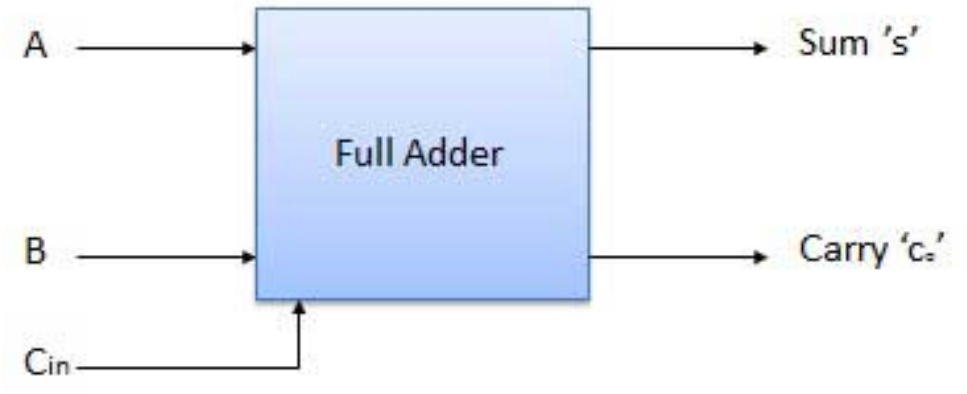


(a)

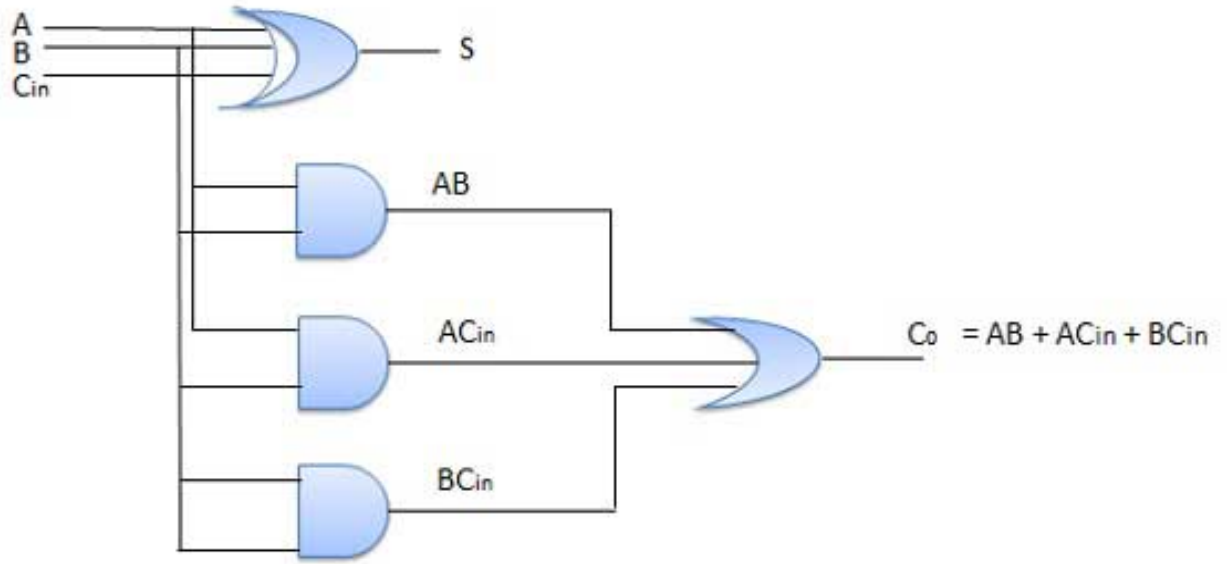


(b)

Somador Completo

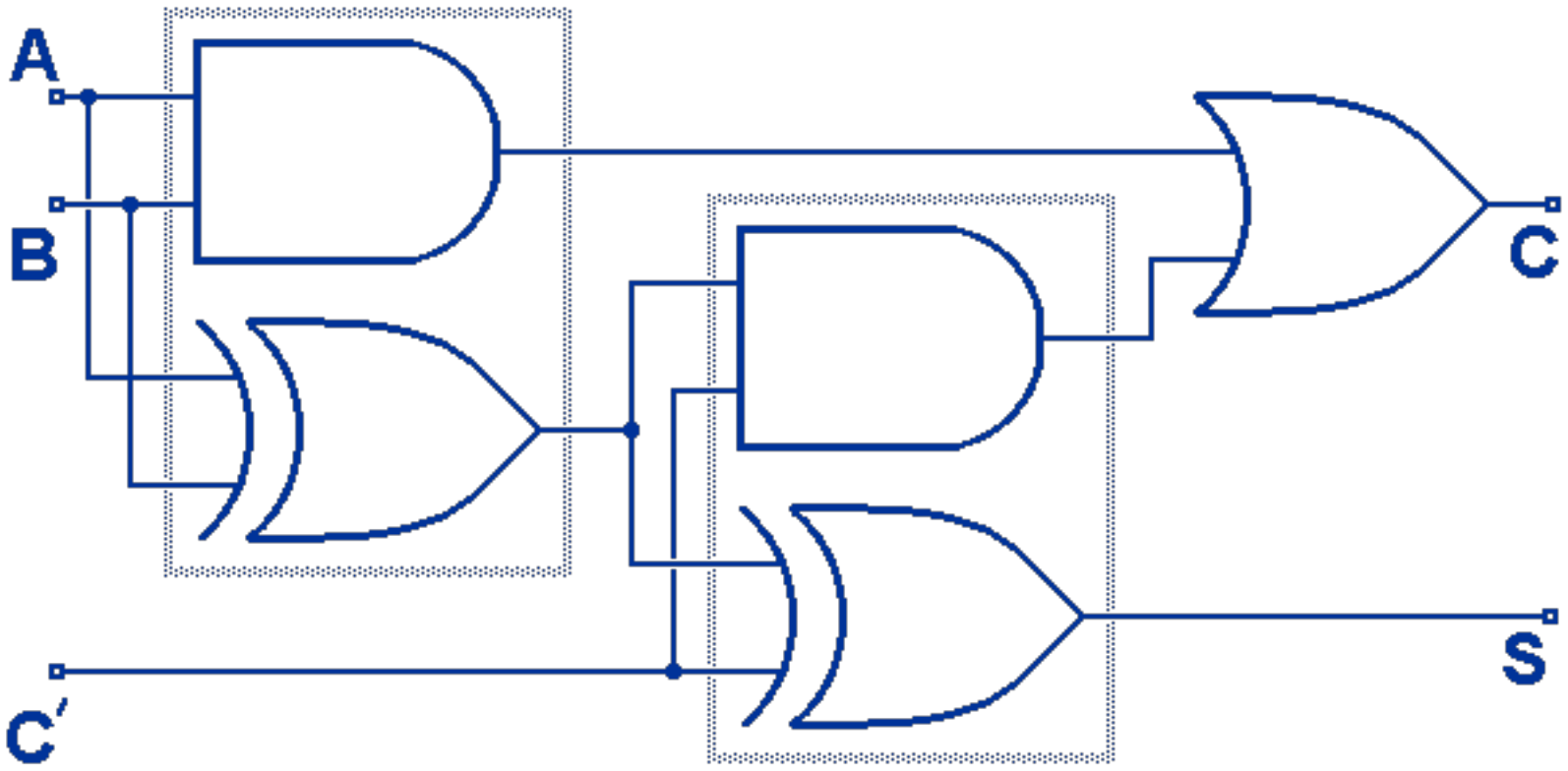


Inputs			Output	
A	B	Cin	S	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Somador Completo

Modo 2: OU podemos implementar a partir de 2 *Half Adders*:



Vejamos como a seguir...

Somador Completo

Modo 2:

Operandos		Carry de entrada	Bit de soma	Carry de saída
A	B	C_{in}	Σ	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$C_{out} = \bar{A}BC_{in} + A\bar{B}C_{in} + AB\overline{C_{in}} + ABC_{in}$$

$$C_{out} = C_{in} \cdot (\bar{A}B + A\bar{B}) + AB(\overline{C_{in}} + C_{in})$$

$$\boxed{C_{out} = C_{in} \cdot (A \oplus B) + AB}$$

$$\Sigma = \bar{A}\bar{B}C_{in} + \bar{A}B\overline{C_{in}} + A\bar{B}\overline{C_{in}} + ABC_{in}$$

$$\Sigma = C_{in} \cdot (\bar{A}\bar{B} + AB) + \overline{C_{in}} \cdot (\bar{A}B + A\bar{B})$$

$$\Sigma = C_{in} \cdot (\overline{A \oplus B}) + \overline{C_{in}} \cdot (A \oplus B)$$

$$X = C_{in}$$

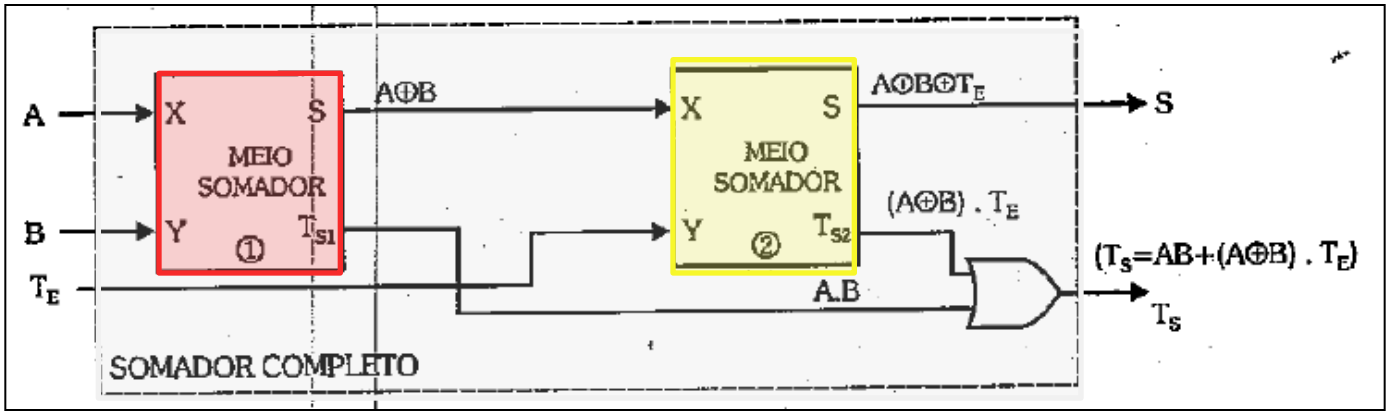
$$Y = (A \oplus B)$$

$$\Sigma = X \oplus Y$$

$$\boxed{\Sigma = C_{in} \oplus (A \oplus B)}$$

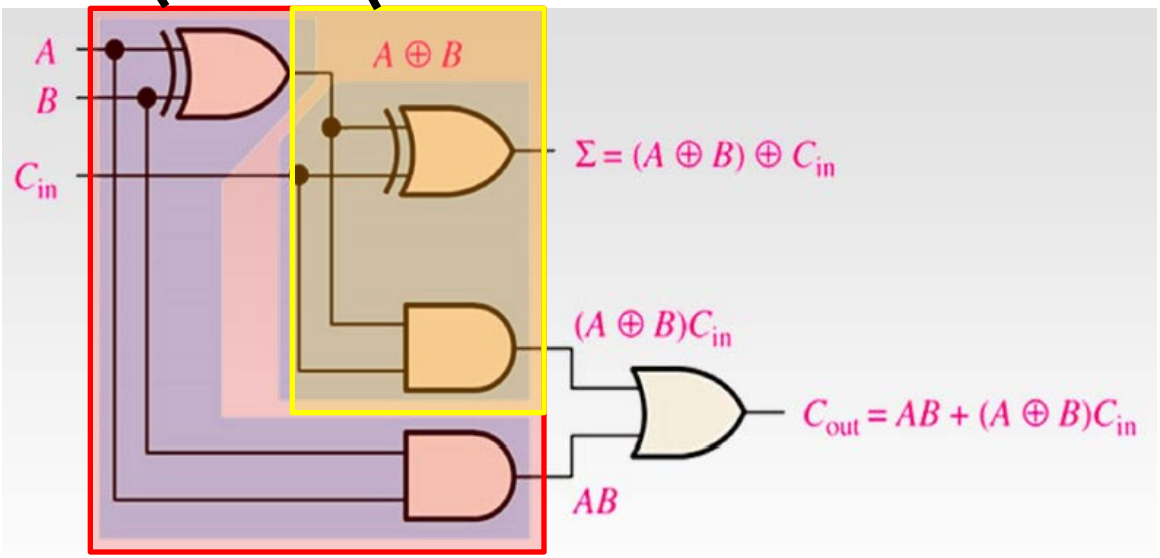
Somador Completo

Modo 2:



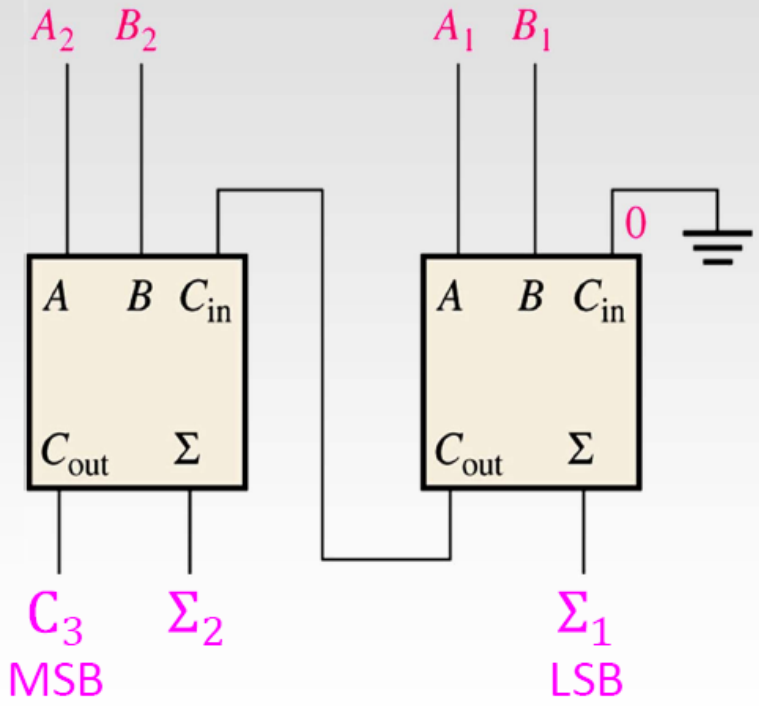
Somador Parcial

Somador Parcial



Somador Propagado de 2 bits

$$\begin{array}{r} A_2A_1 \\ +\ B_2B_1 \\ \hline C_3\Sigma_2\Sigma_1 \end{array}$$



Somador Propagado de 4 bits

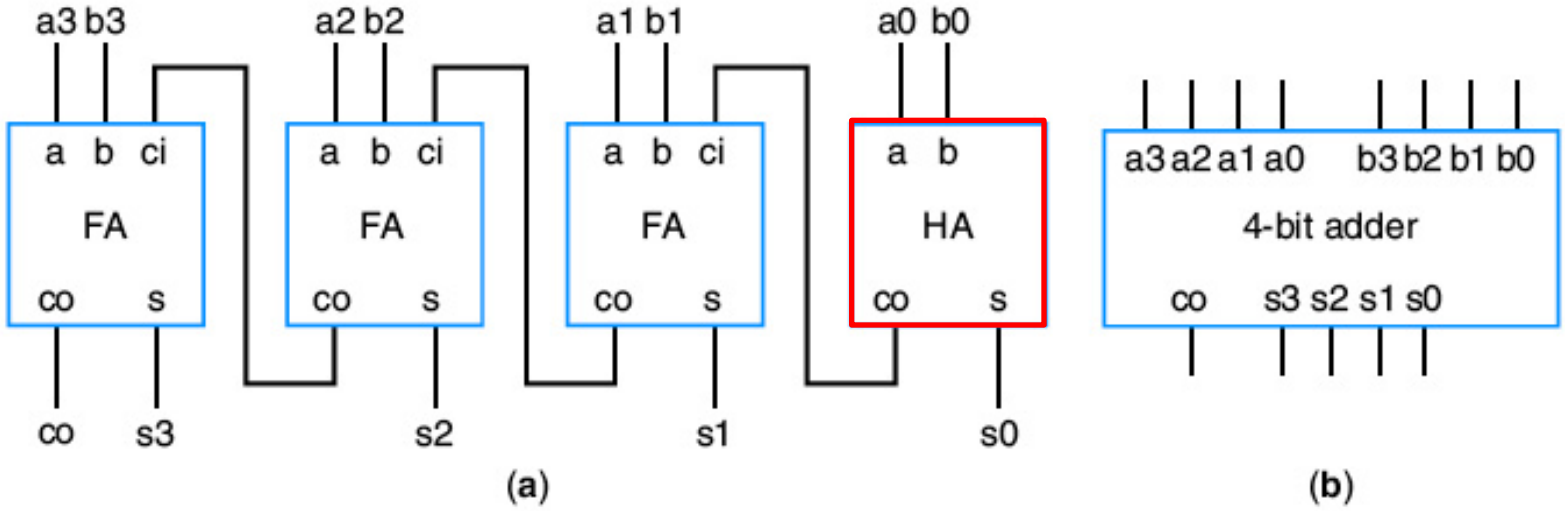
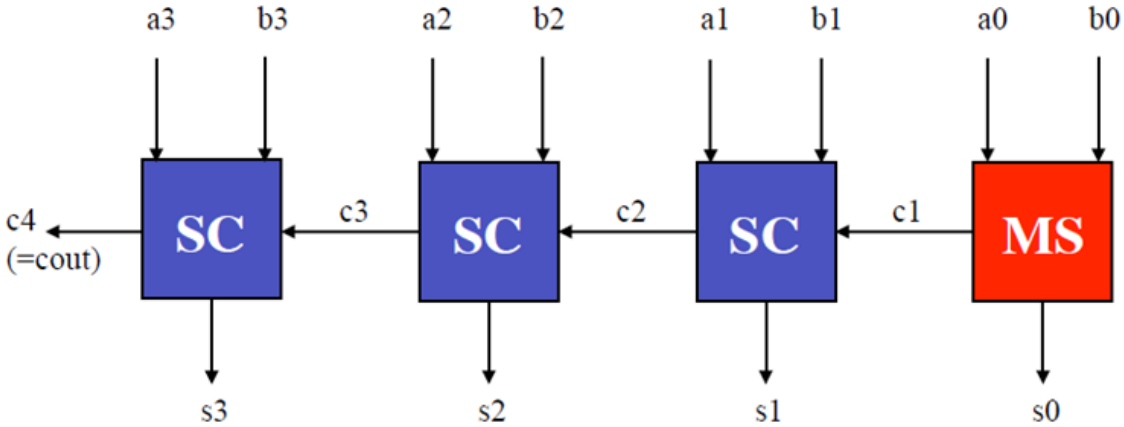


Diagrama de Blocos (Nível Lógico)



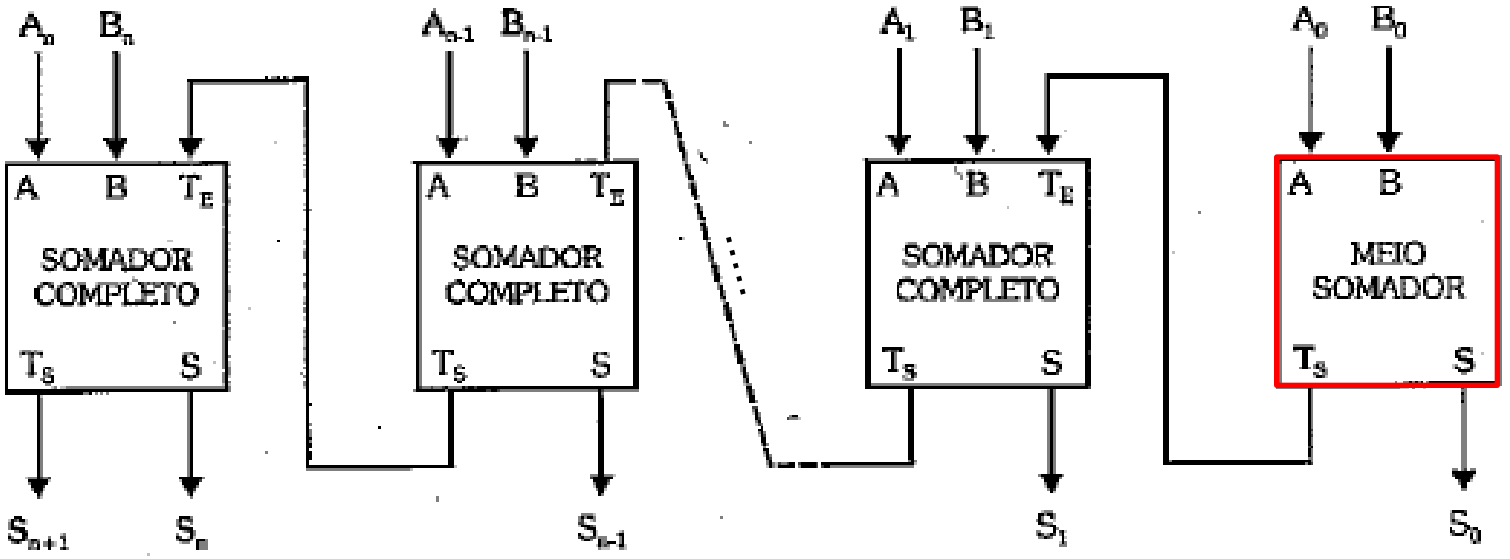
Versão 1

Somador Propagado de N bits

- O somador parcial só pode ser usado na coluna menos significativa da operação (posição 0)
- O somador parcial é insuficiente para as colunas mais significativas, pois não leva em consideração o transporte (vem um) oriundo da posição anterior.

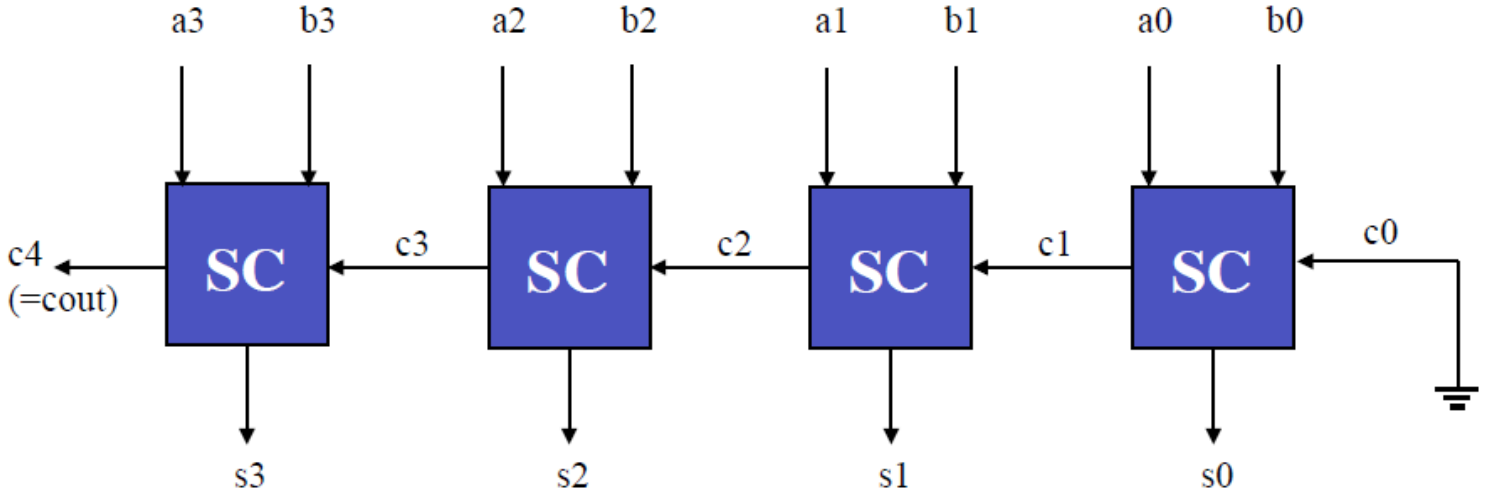
$$\begin{array}{rr} & C_{in}\ \ C_{in}\ \cdots C_{in} \\ & A_{n-1}A_{n-2}\cdots A_1A_0 \\ + & B_{n-1}B_{n-2}\cdots B_1B_0 \\ \hline C_{out} & S_{n-1}\ S_{n-2}\cdots\ S_1\ S_0 \end{array}$$

Somador, desconsiderando o vem-1 inicial.

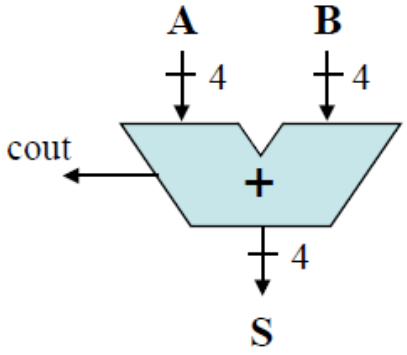


Somador Propagado com Vem-1 Inicial Nulo

Diagrama de Blocos (Nível Lógico): versão 2



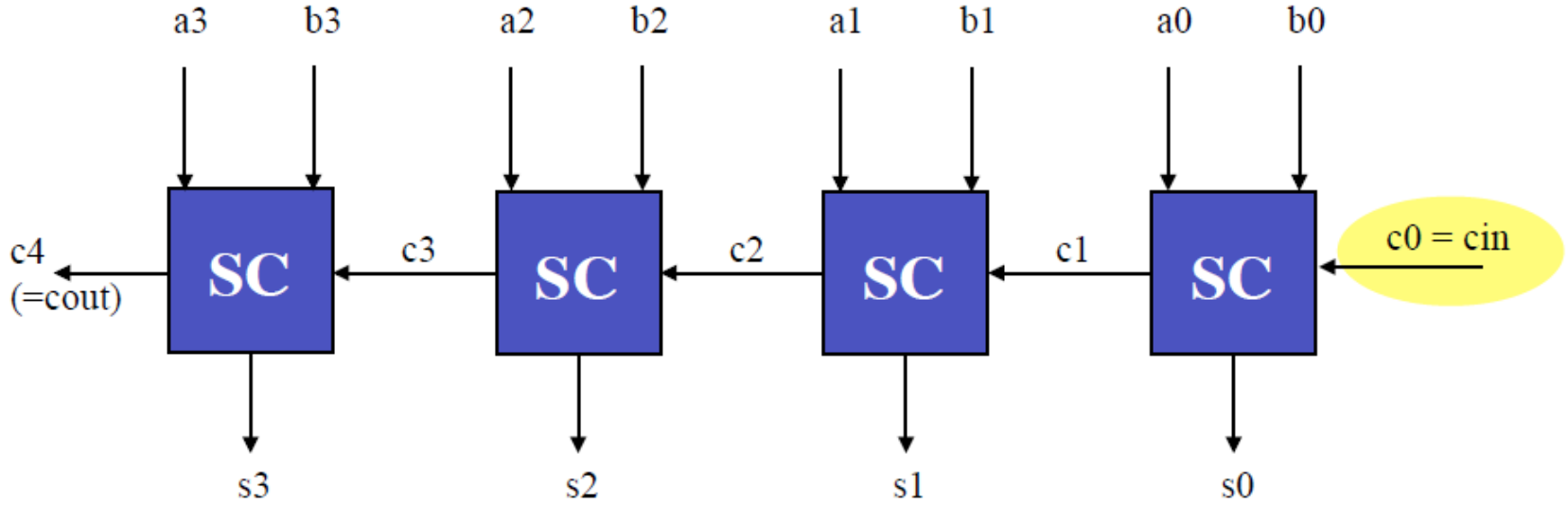
Símbolo no Nível RT



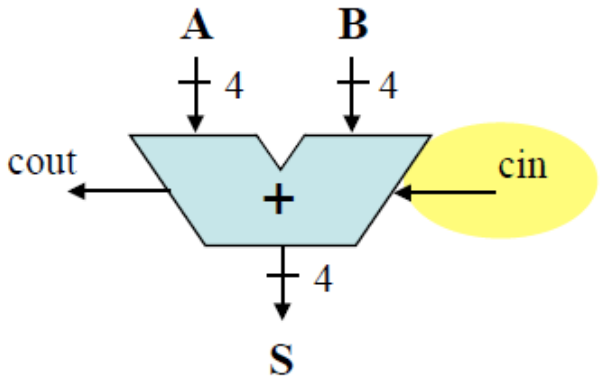
A ideia empregada nesta versão pode ser generalizada para um somador de N bits.

Somador Propagado com Vem-1 Inicial Não-Nulo

Diagrama de Blocos (Nível Lógico): versão 3



Símbolo no Nível RT



A ideia empregada nesta versão pode ser generalizada para um somador de N bits.

MÓDULOS ARITMÉTICOS:

SUBTRATORES

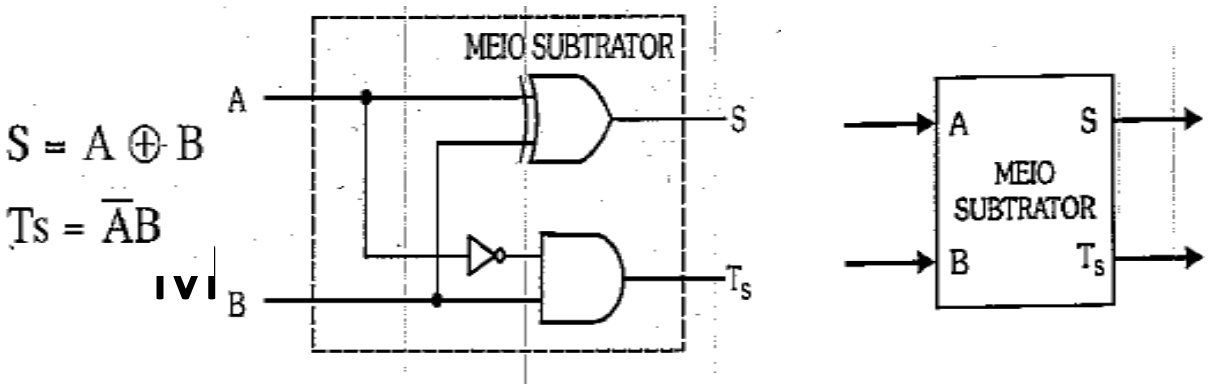
Subtrator Parcial (ou Meio Subtrator)

Um subtrator de N bits é um componente de bloco operacional que toma duas entradas binárias A e B e produz um resultado S na saída

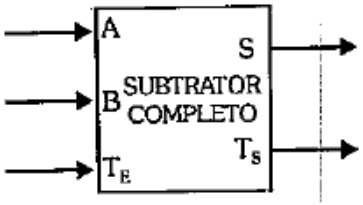
A	B	S	Ts
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$S = A \oplus B$

$Ts = \overline{A}B$



Subtrator Completo



A	B	T_E	S	T_S
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

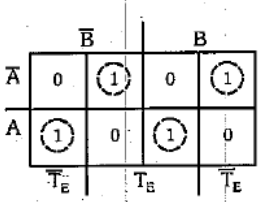
As expressões características extraídas da tabela são:

$S = \overline{A}\,\overline{B}T_E + \overline{A}B\overline{T_E} + A\overline{B}\overline{T_E} + ABT_E$

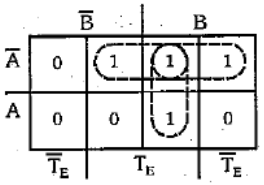
$Ts = \overline{A}\,\overline{B}T_E + \overline{A}B\overline{T_E} + A\overline{B}\overline{T_E} + ABT_E$

Vamos simplificar estas expressões:

S:

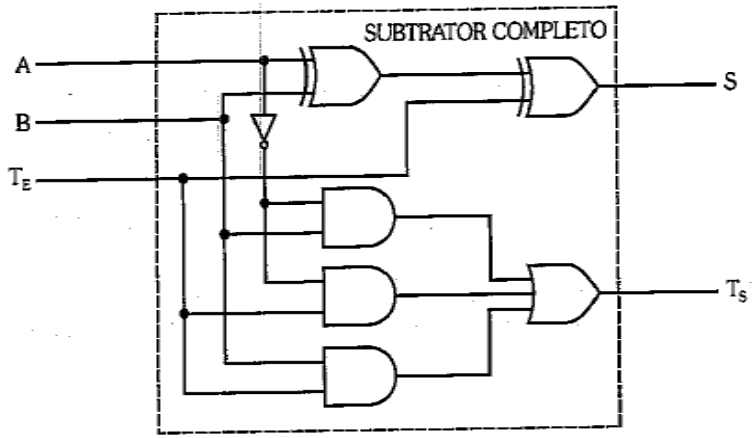


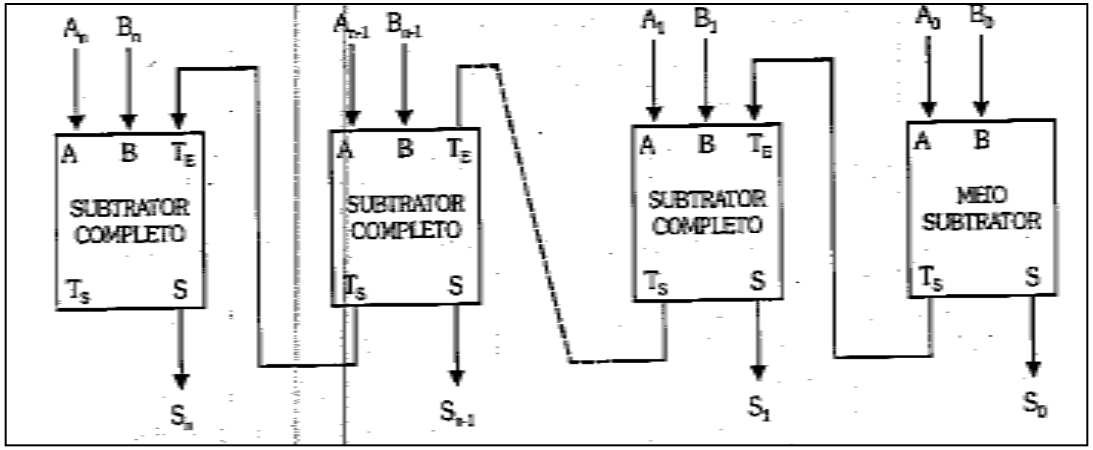
T_S:



(a) $S = A \oplus B \oplus T_E$

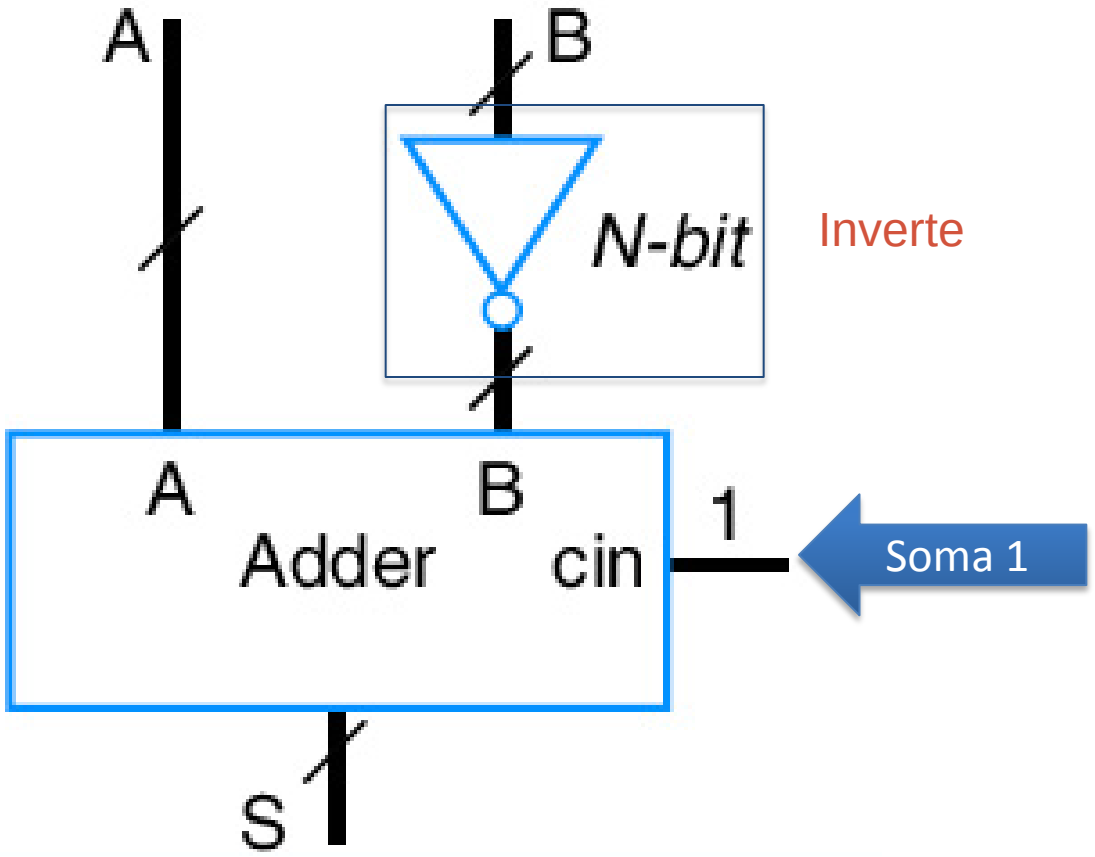
(b) $Ts = \overline{A}B + \overline{A}T_E + BT_E$





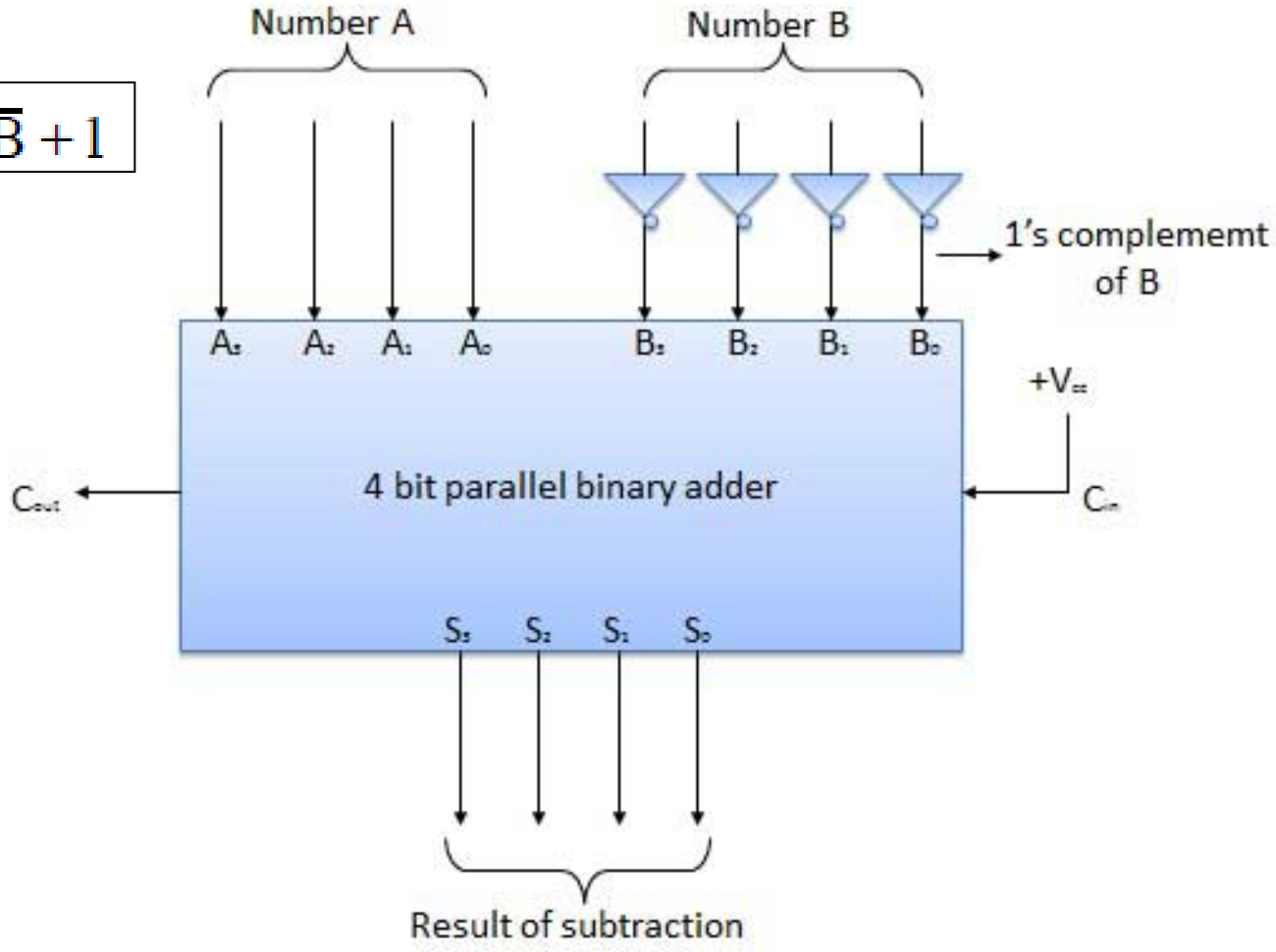
Subtrator (via Complemento de 2)

$$A - B = A + \overline{B} + 1$$



Subtrator de N bits (via Complemento de 2)

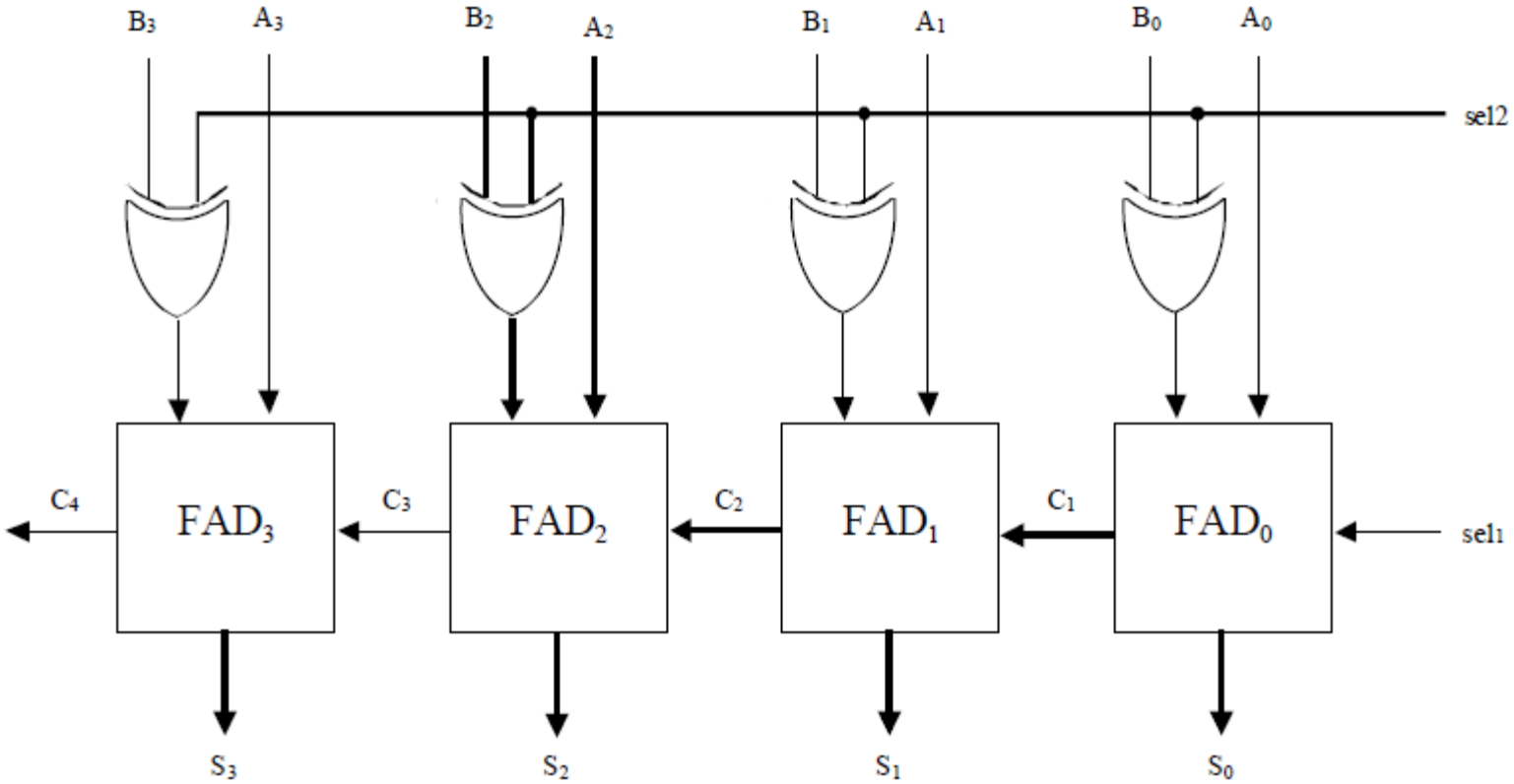
$$A - B = A + \overline{B} + 1$$



Somador e Subtrator Combinados

sel_2	sel_1	operação	descrição
0	0	$S = A + B + 0$	adiciona A e B ($S = A + B$)
0	1	$S = A + B + 1$	adiciona A e B incrementado ($S = A + B + 1$)
1	0	$S = A + \overline{B} + 0$	subtrai B decrementado de A ($S = A - B - 1$)
1	1	$S = A + \overline{B} + 1$	subtrai B de A ($S = A - B$)

$$A - B = A + \overline{B} + 1$$



Somador/subtrator de 4 bits.

Somador e Subtrator Combinados

- A subtração em complemento 2 pode ser convertida em adição.
- Exemplo com 4 bits:

$$A_3A_2A_1A_0 - B_3B_2B_1B_0$$

$$A_3A_2A_1A_0 + (-B_3B_2B_1B_0)$$

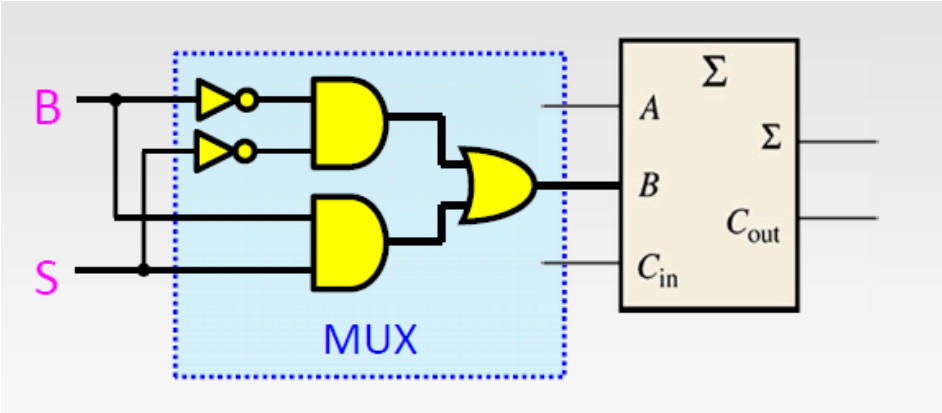
$$A_3A_2A_1A_0 + (\overline{B_3}\ \overline{B_2}\ \overline{B_1}\ \overline{B_0} + 1)$$

- É desejável projetar um único circuito que some ou subtraia conforme um seletor seja ativado ou desativado.

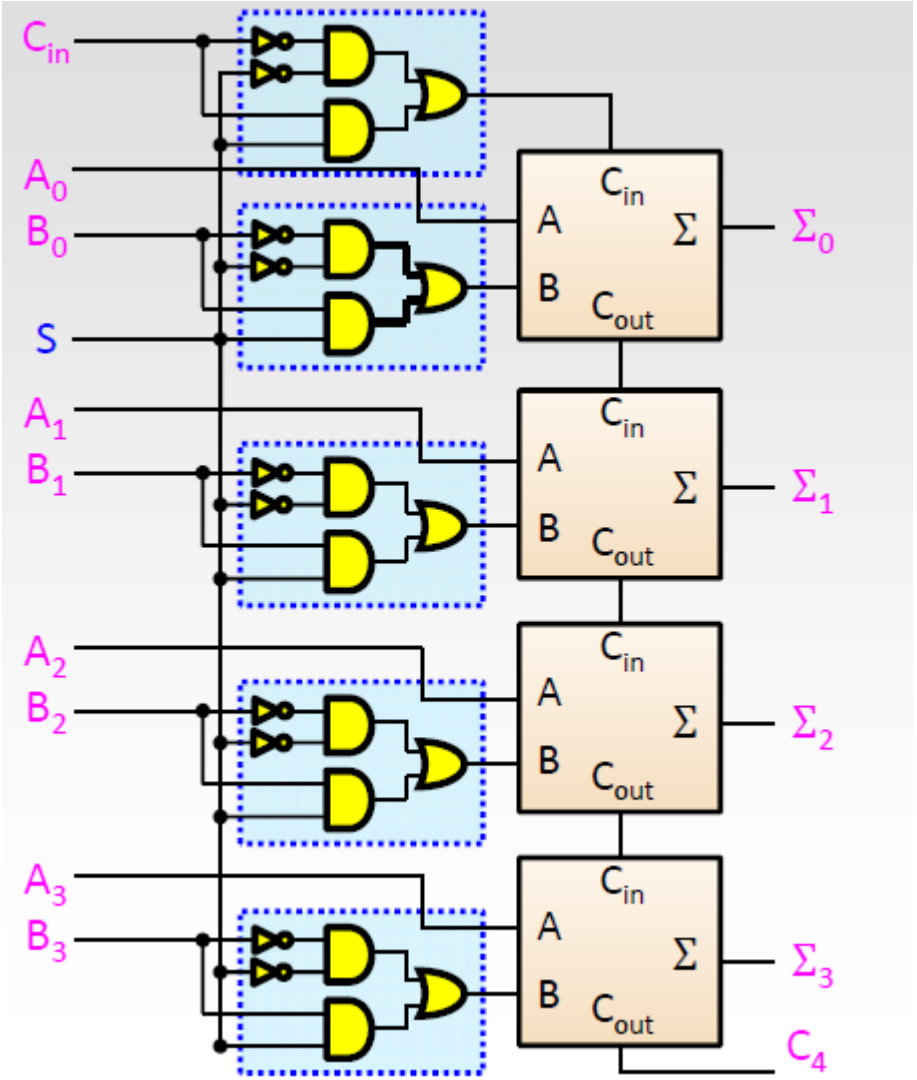
seletor ativado para soma (S=1)	seletor ativado para subtração (S=0)
• Entradas B_i permanecem as mesmas • Na posição LSB, $C_{in} = 0$	• Entradas B_i devem ser negadas • Na posição LSB, $C_{in} = 1$

Somador e Subtrator Combinados

1 bit



4 bits

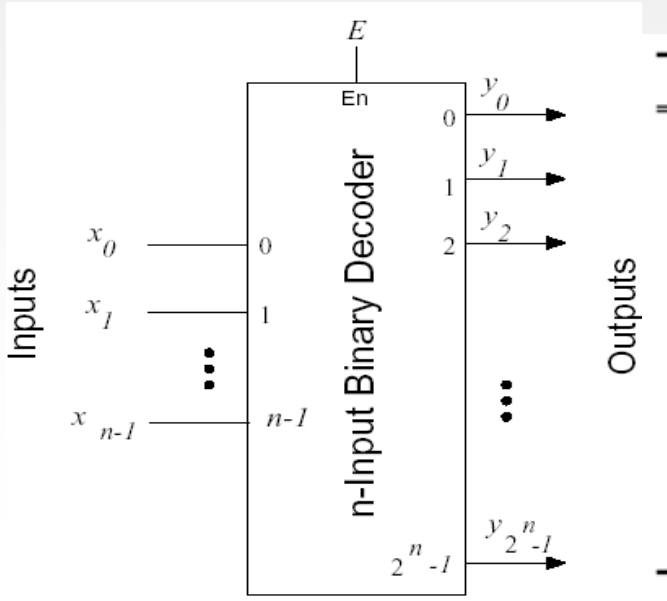


MÓDULOS PADRÃO-COMBINACIONAIS

DECODIFICADORES (*DECODERS*) / CODIFICADORES (*ENCODERS*)

Decodificadores

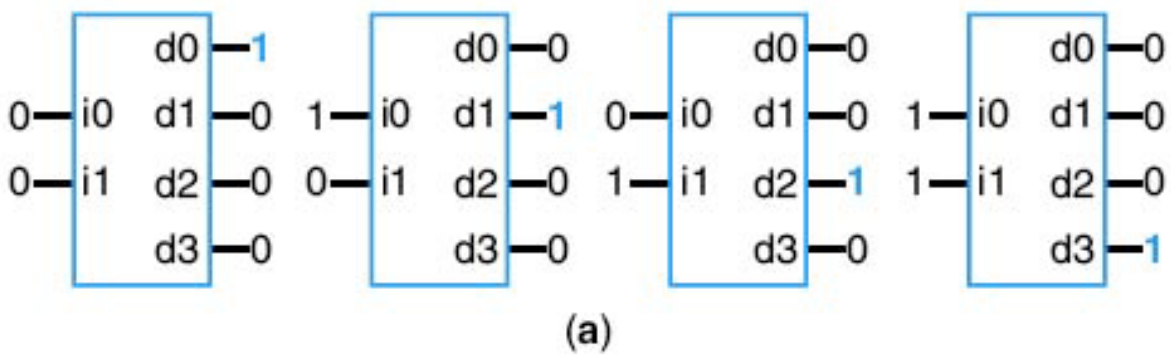
- Um decodificador é um circuito que recebe um conjunto de entradas que representam um número binário e ativa apenas a saída correspondente ao valor recebido.
- Para cada configuração de bits que aparece na entrada, haverá uma e somente uma linha de saída ativa.
- Geralmente, o circuito tem n vias de entrada e 2^n vias de saída.



E	x_2	x_1	x_0	x	y_7	y_6	y_5	y_4	y_3	y_2	y_1	y_0
1	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	1	0	0	0	0	0	0	1	0
1	0	1	0	2	0	0	0	0	0	1	0	0
1	0	1	1	3	0	0	0	0	1	0	0	0
1	1	0	0	4	0	0	0	1	0	0	0	0
1	1	0	1	5	0	0	1	0	0	0	0	0
1	1	1	0	6	0	1	0	0	0	0	0	0
1	1	1	1	7	1	0	0	0	0	0	0	0
0	-	-	-	-	0	0	0	0	0	0	0	0

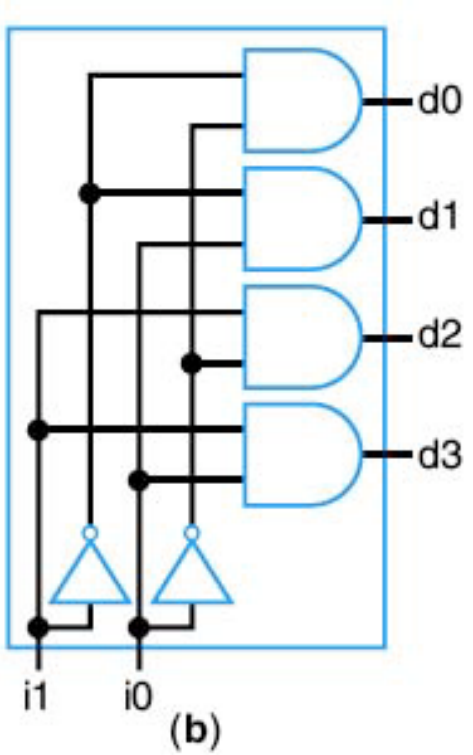
Decodificadores

DECO (n x 2^n) → Decodifica um número binário de n bits na entrada colocando somente uma das 2^n saídas em 1



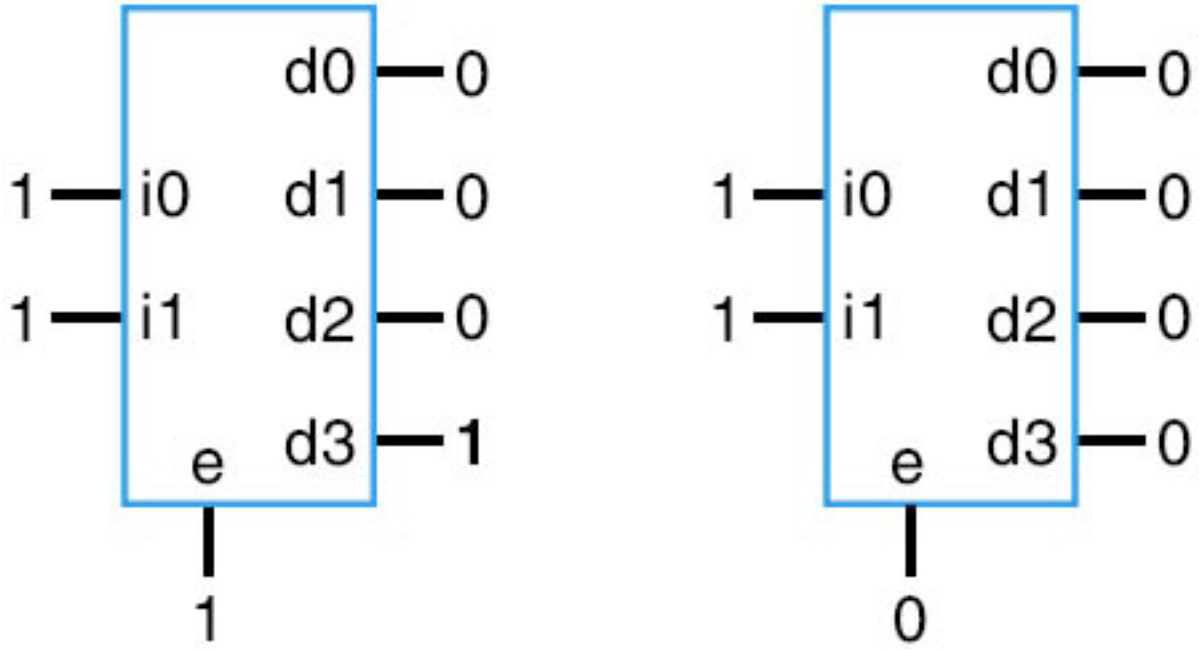
DECO (2 X 4)

Como são as equações para d0, d1, d2, d3?



Entrada Enable

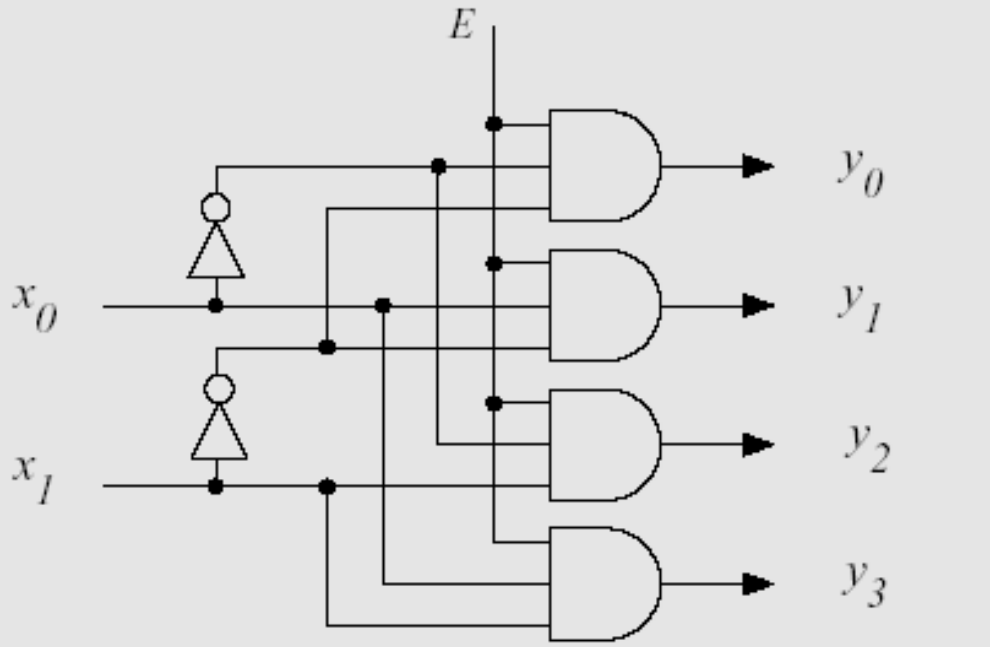
- Habilita o funcionamento de um circuito.
- No caso do decodificador:



Circuito Lógico

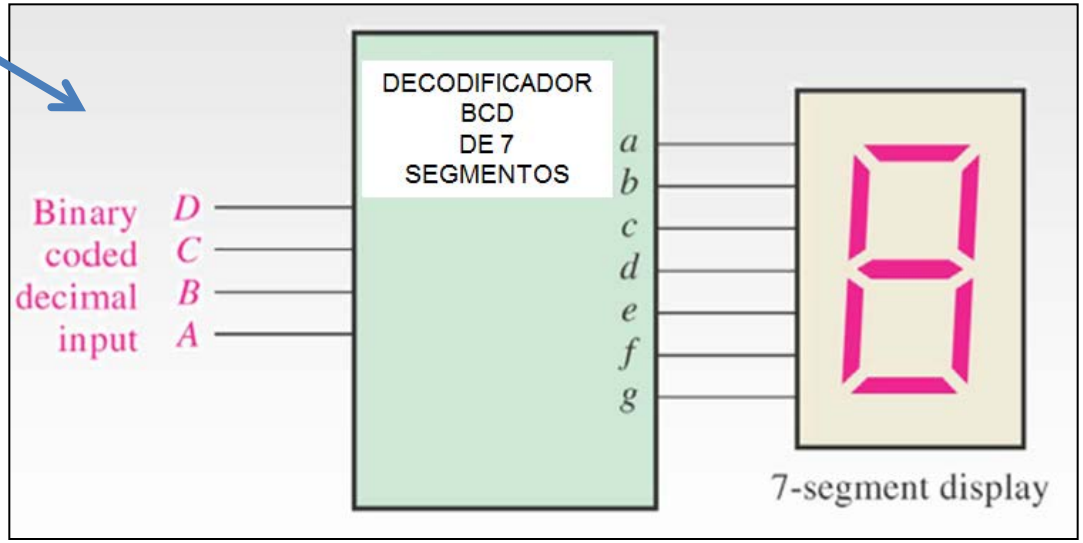
- Exemplo de Circuito de Decodificador com 2 entradas e 4 saídas : (2×4)

$y_0 = x_1' x_0' E \quad y_1 = x_1' x_0 E \quad y_2 = x_1 x_0' E \quad y_3 = x_1 x_0 E$

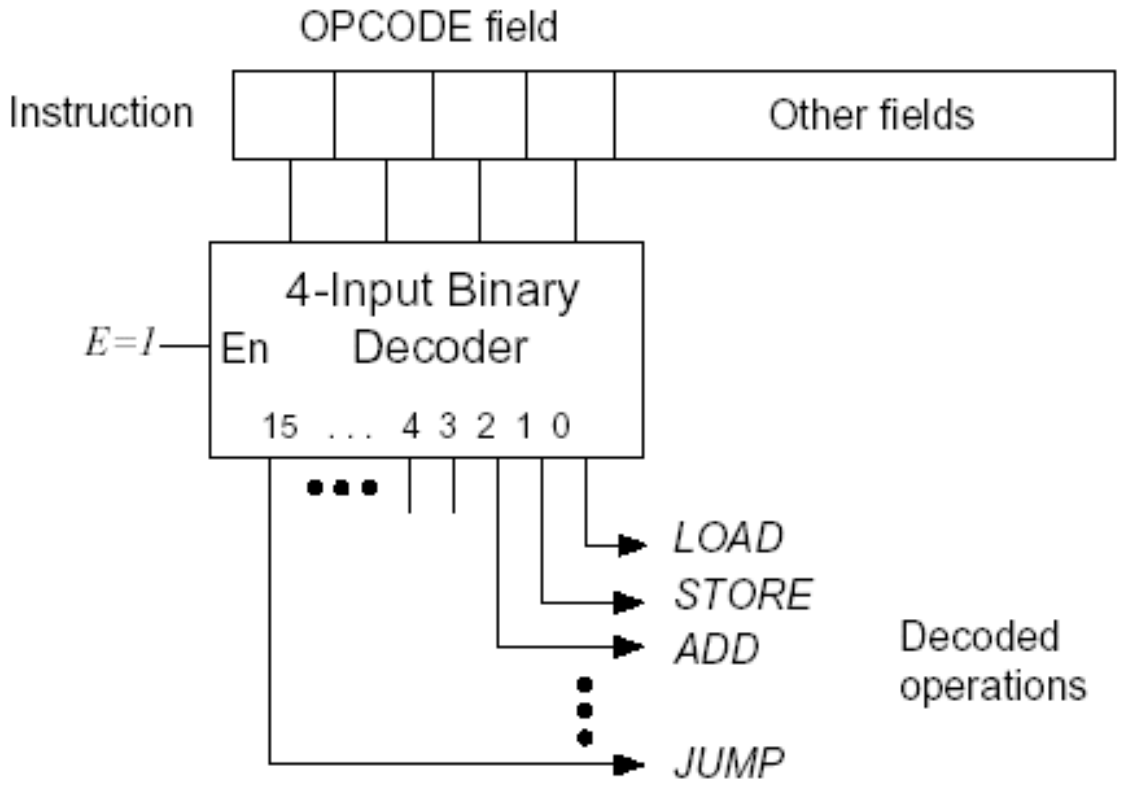


Exemplos de Decodificadores (Decoders)

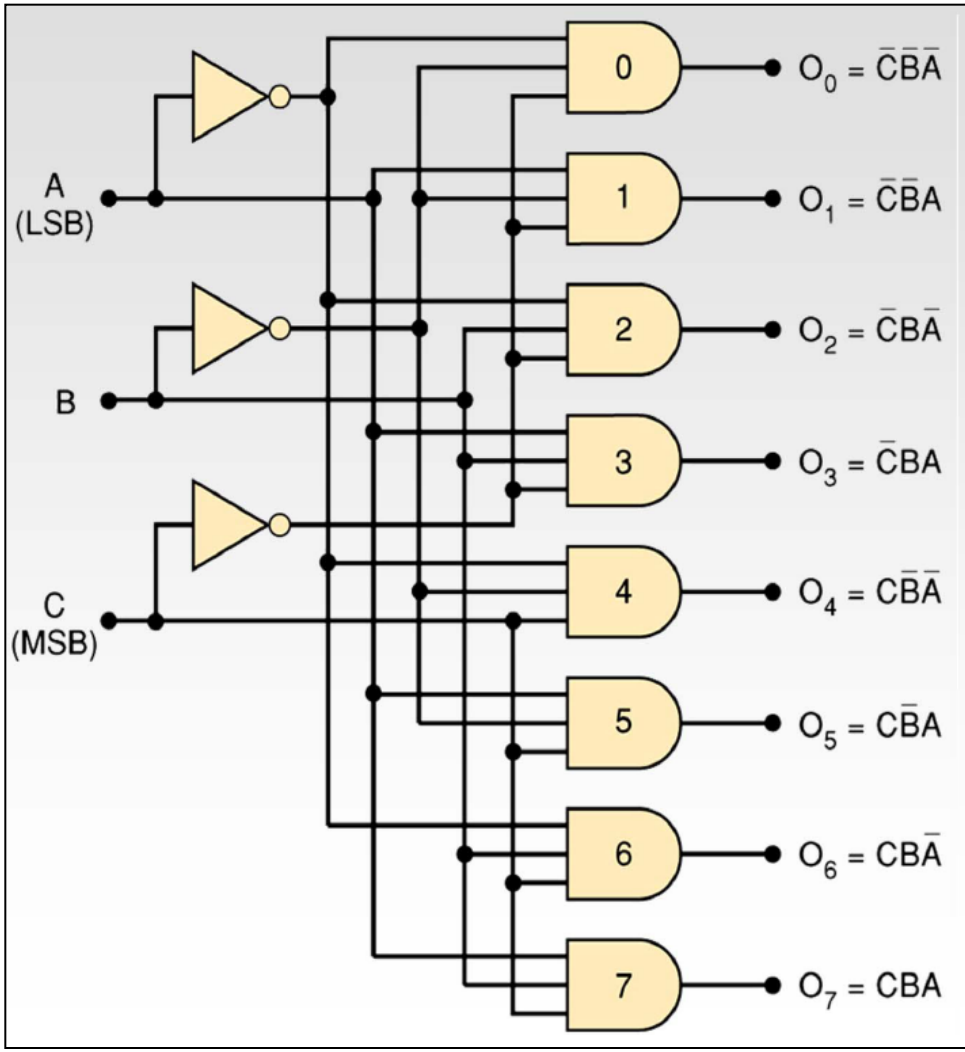
- Conversores de Códigos.
- Conversores de BCD para Displays de 7 Segmentos



Decodificadores: Outro Exemplo



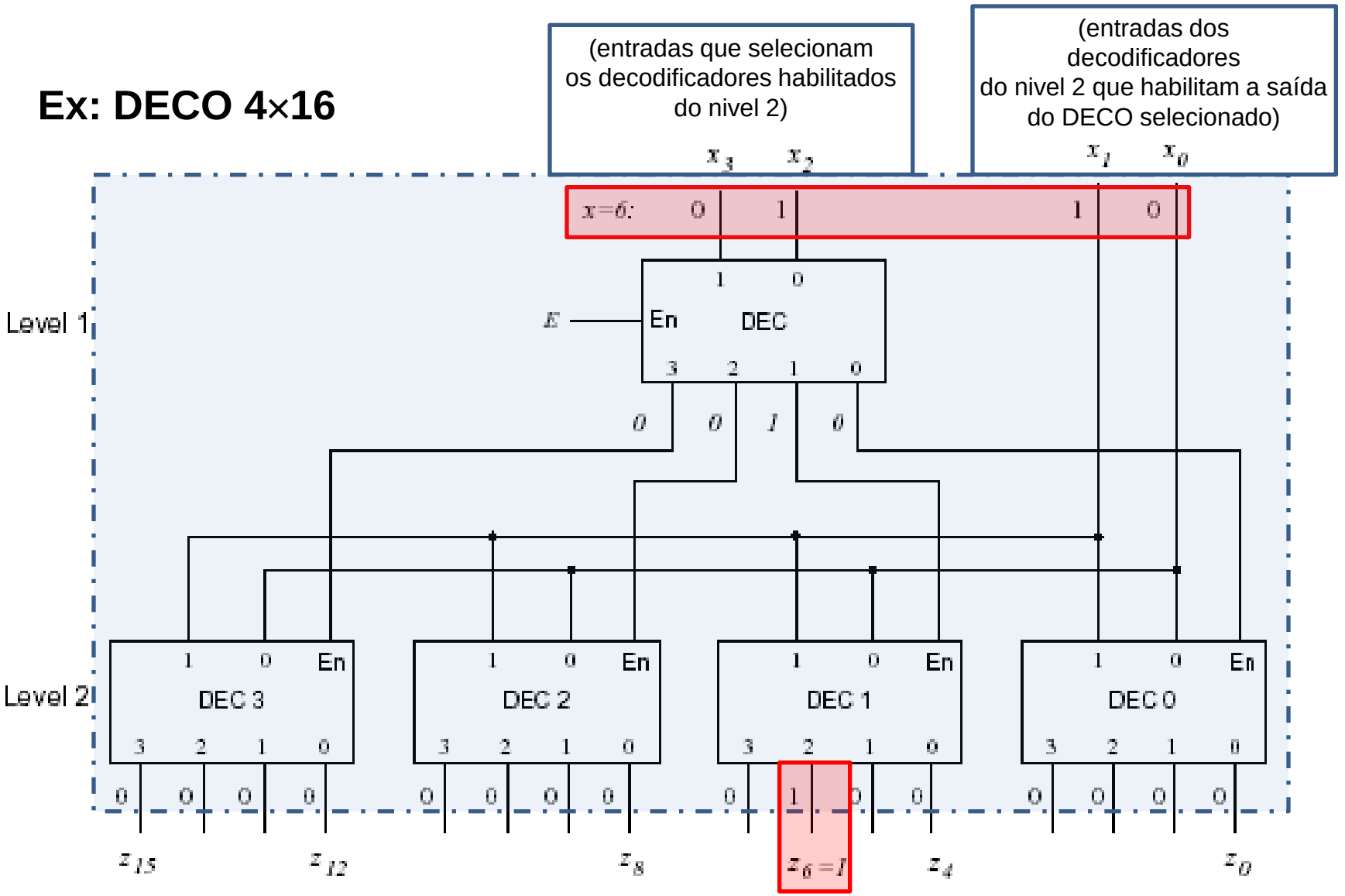
Exemplo: Decodificador de 3 para 8 bits



C	B	A	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

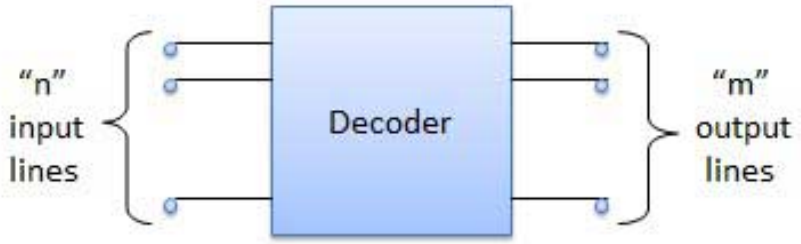
Decodificadores em Árvore

Ex: DECO 4×16

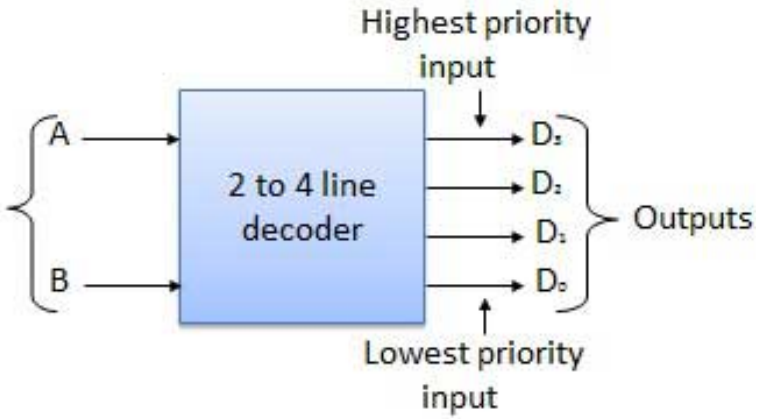


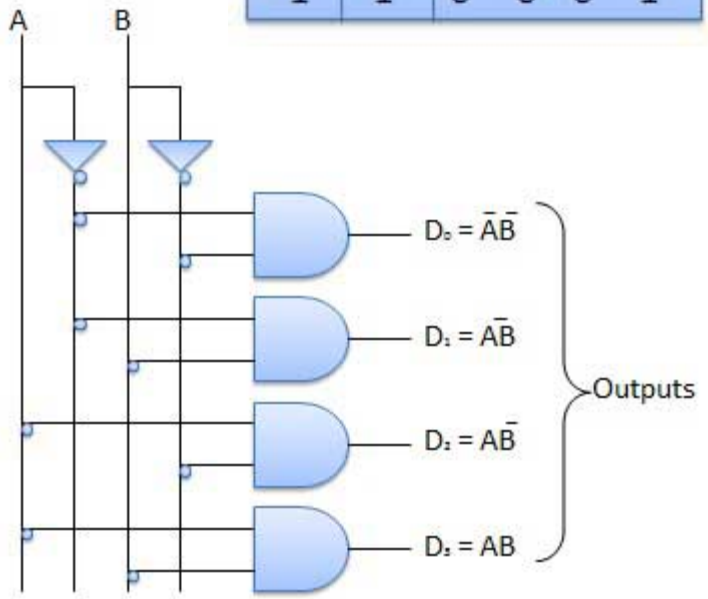
Decodificadores (DECODERS)

RESUMO:



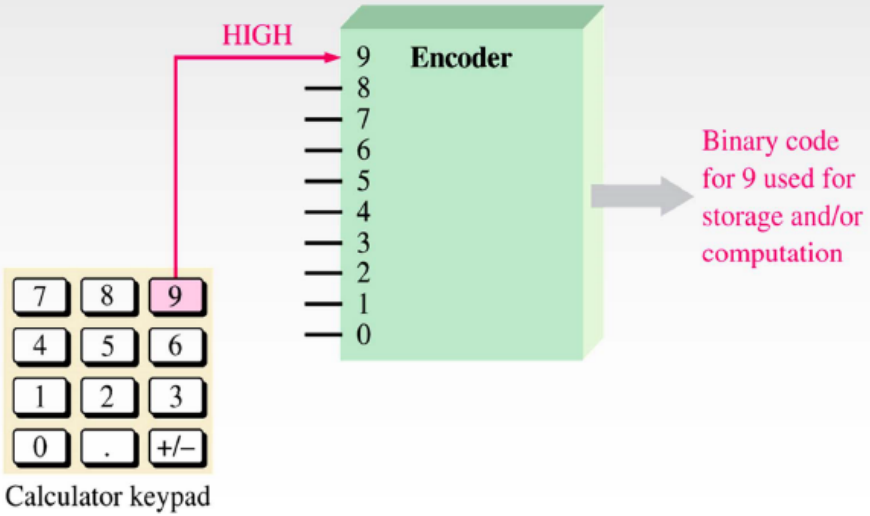
Inputs		Output			
A	B	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
0	1	0	0	1	0
1	1	0	0	0	1

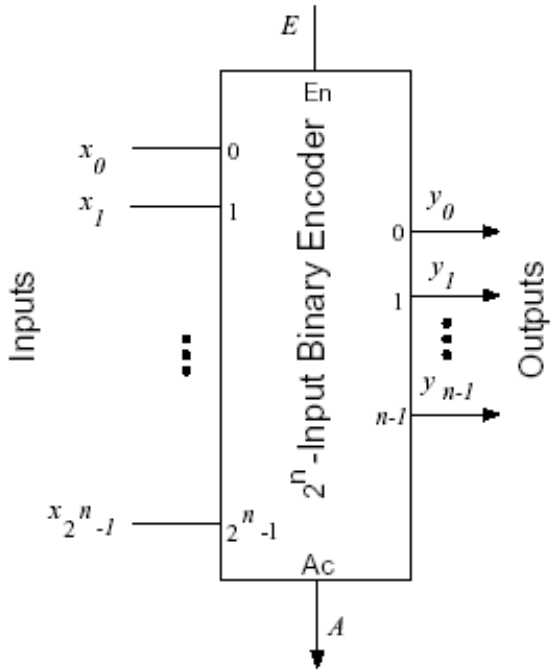




Codificadores

- São circuitos combinatórios que têm um certo número de linhas de entrada, em que somente uma linha é ativada por vez, produzindo um código de saída de N bits, a depender de qual entrada for ativada.





E	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0	y	y_2	y_1	y_0	A
1	0	0	0	0	0	0	0	1	0	0	0	0	1
1	0	0	0	0	0	0	1	0	1	0	0	1	1
1	0	0	0	0	0	1	0	0	2	0	1	0	1
1	0	0	0	0	1	0	0	0	3	0	1	1	1
1	0	0	0	1	0	0	0	0	4	1	0	0	1
1	0	0	1	0	0	0	0	0	5	1	0	1	1
1	0	1	0	0	0	0	0	0	6	1	1	0	1
1	1	0	0	0	0	0	0	0	7	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	-	-	-	-	-	-	-	-	0	0	0	0	0

Codificadores

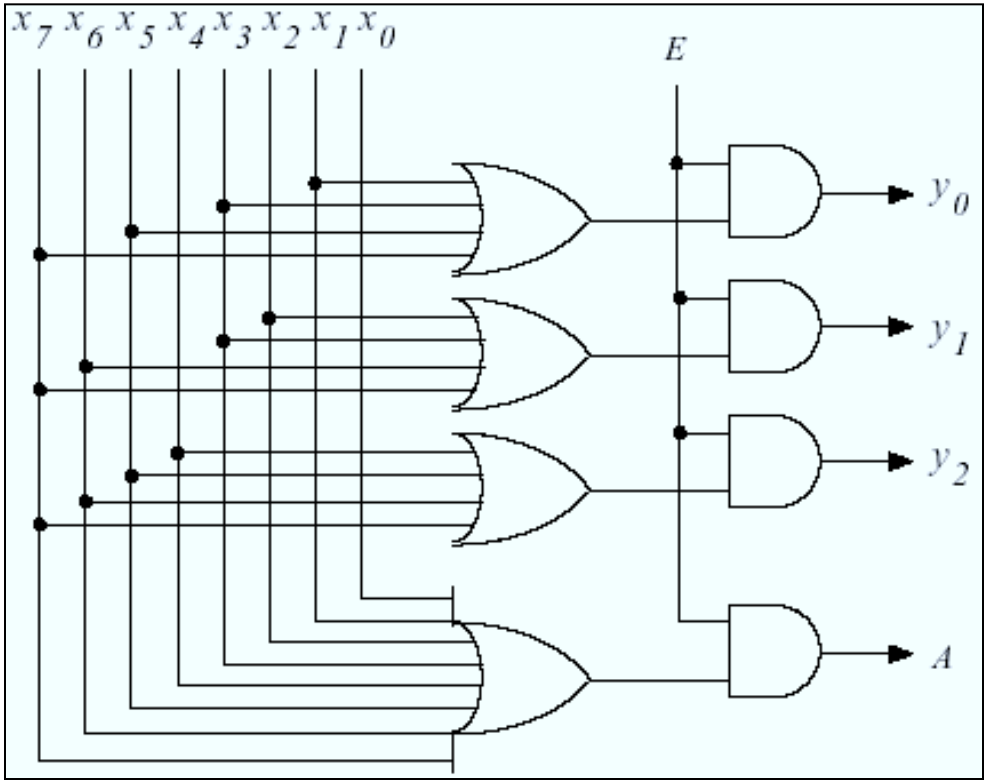
E	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0	y	y_2	y_1	y_0	A
1	0	0	0	0	0	0	0	1	0	0	0	0	1
1	0	0	0	0	0	0	1	0	1	0	0	1	1
1	0	0	0	0	0	1	0	0	2	0	1	0	1
1	0	0	0	0	1	0	0	0	3	0	1	1	1
1	0	0	0	1	0	0	0	0	4	1	0	0	1
1	0	0	1	0	0	0	0	0	5	1	0	1	1
1	0	1	0	0	0	0	0	0	6	1	1	0	1
1	1	0	0	0	0	0	0	0	7	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	-	-	-	-	-	-	-	-	0	0	0	0	0

$$y_0 = E \cdot (x_1 + x_3 + x_5 + x_7)$$

$$y_1 = E \cdot (x_2 + x_3 + x_6 + x_7)$$

$$y_2 = E \cdot (x_4 + x_5 + x_6 + x_7)$$

$$A = E \cdot (x_0 + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7)$$

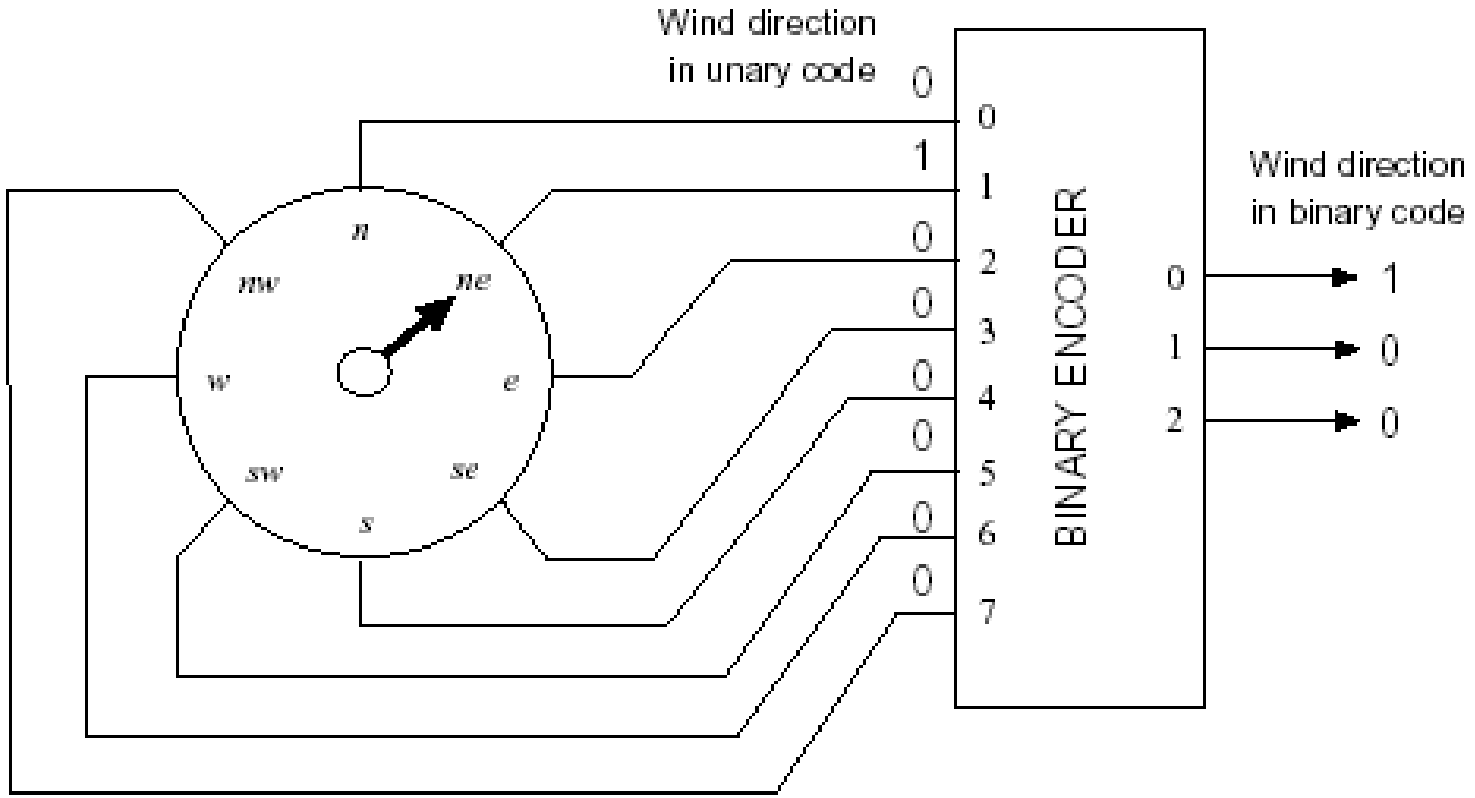


Exemplos de Codificadores (ENCODERS)

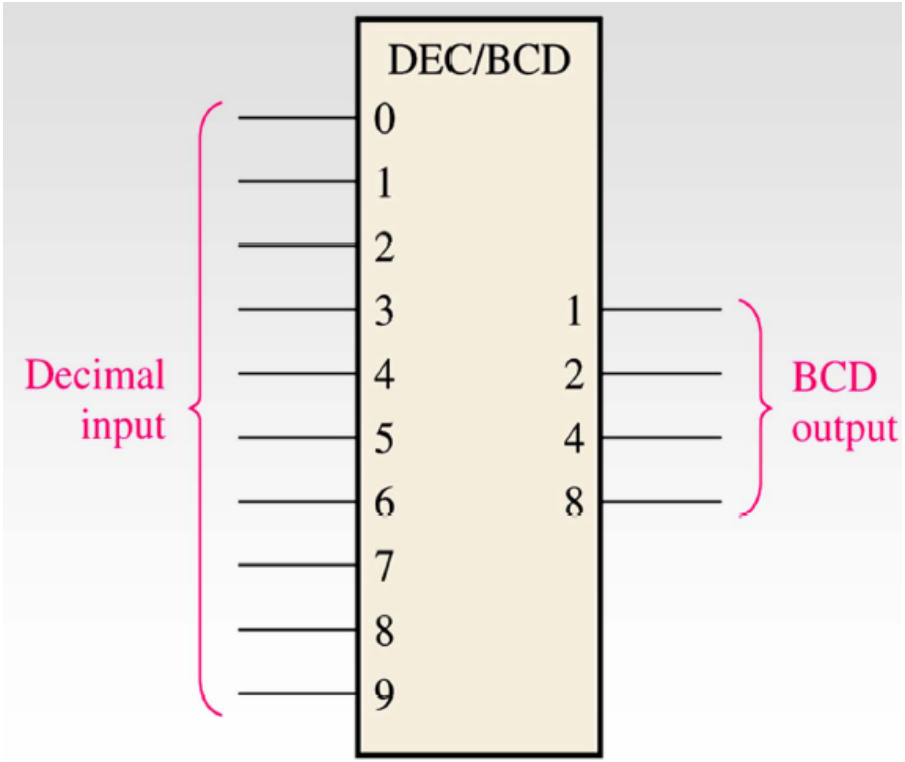
- Codificadores de Prioridades
- Codificador Decimal para BCD
- Codificador Decimal para Binário
- Codificador de Binário para Gray
- Codificador de Binário para BCD…

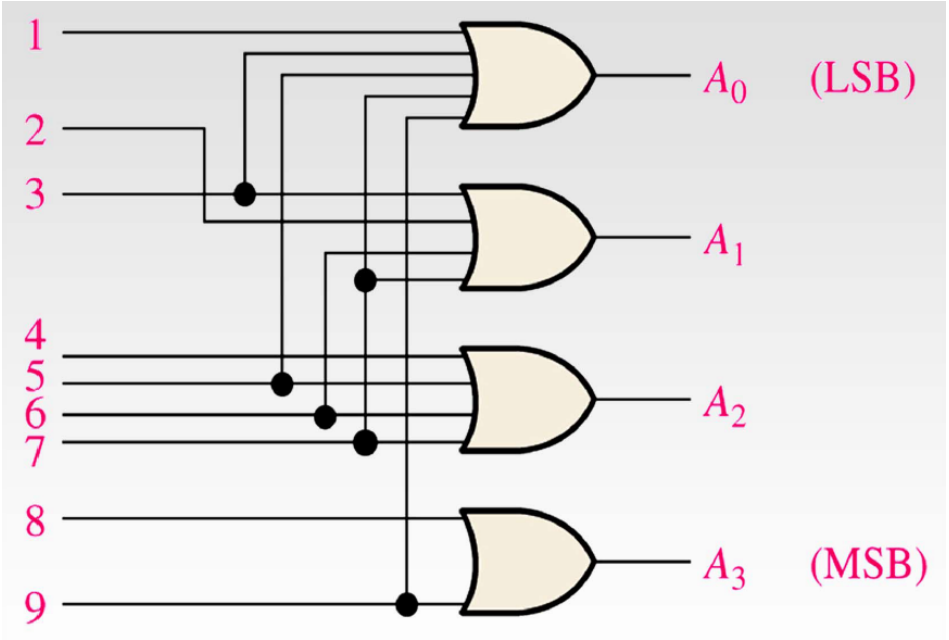
Codificadores: Exemplo

- Exemplo de uso: Determinação da direção de vento.



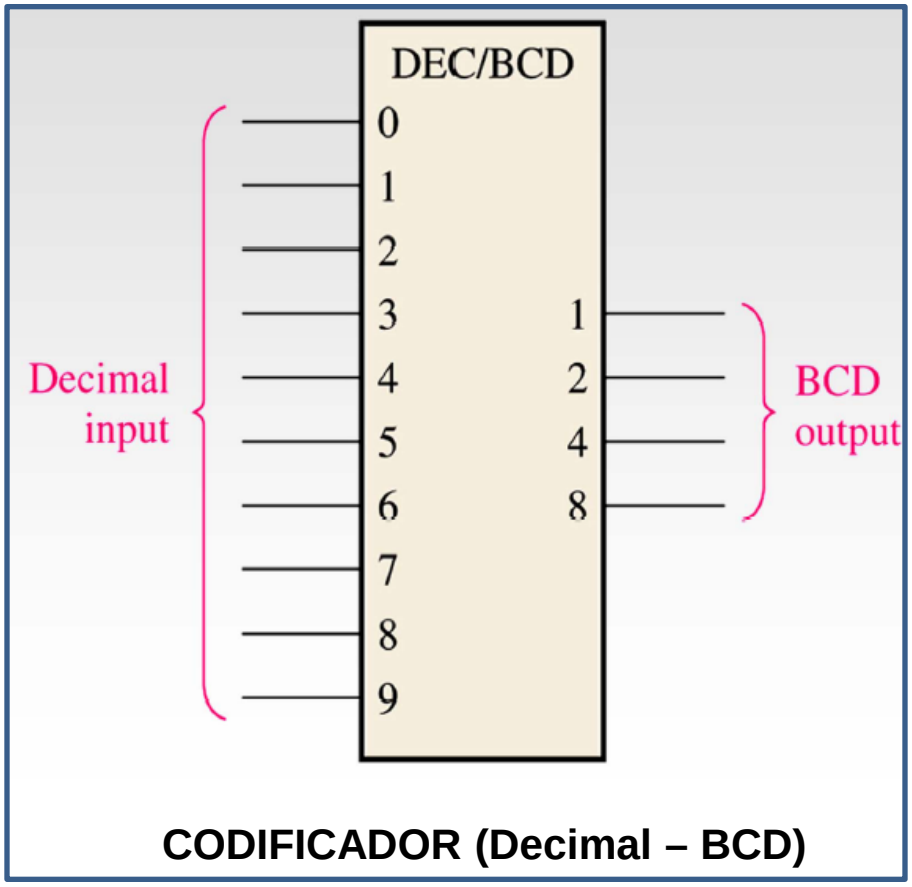
Exemplo Codificador Decimal/BCD



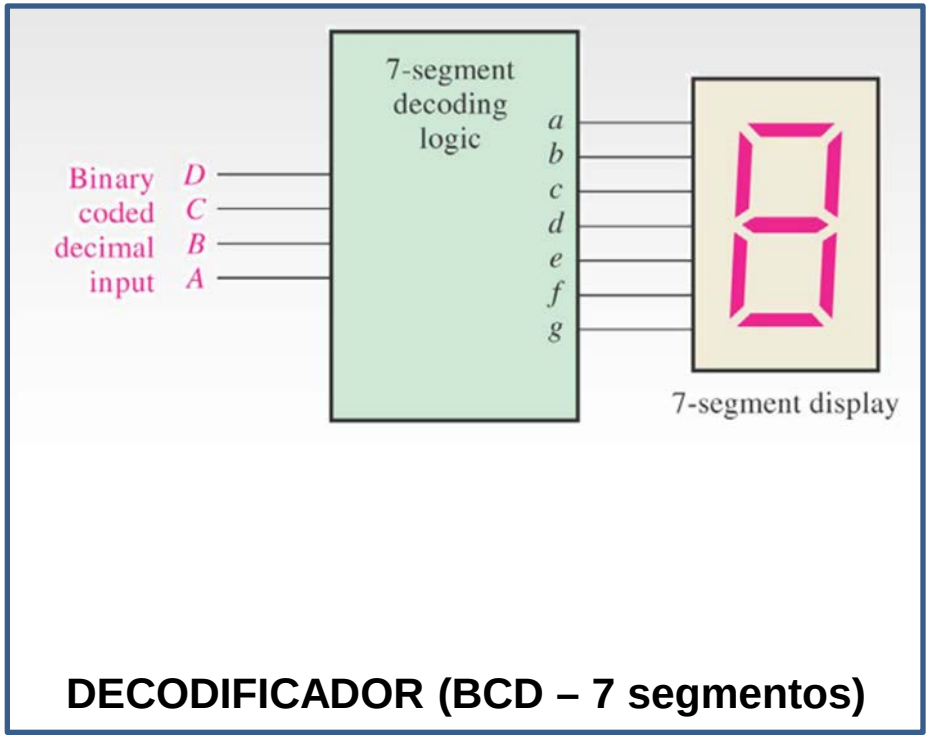


CODIFICADOR (Decimal → BCD)

Uso em Conjunto (ENCODER / DECODER)

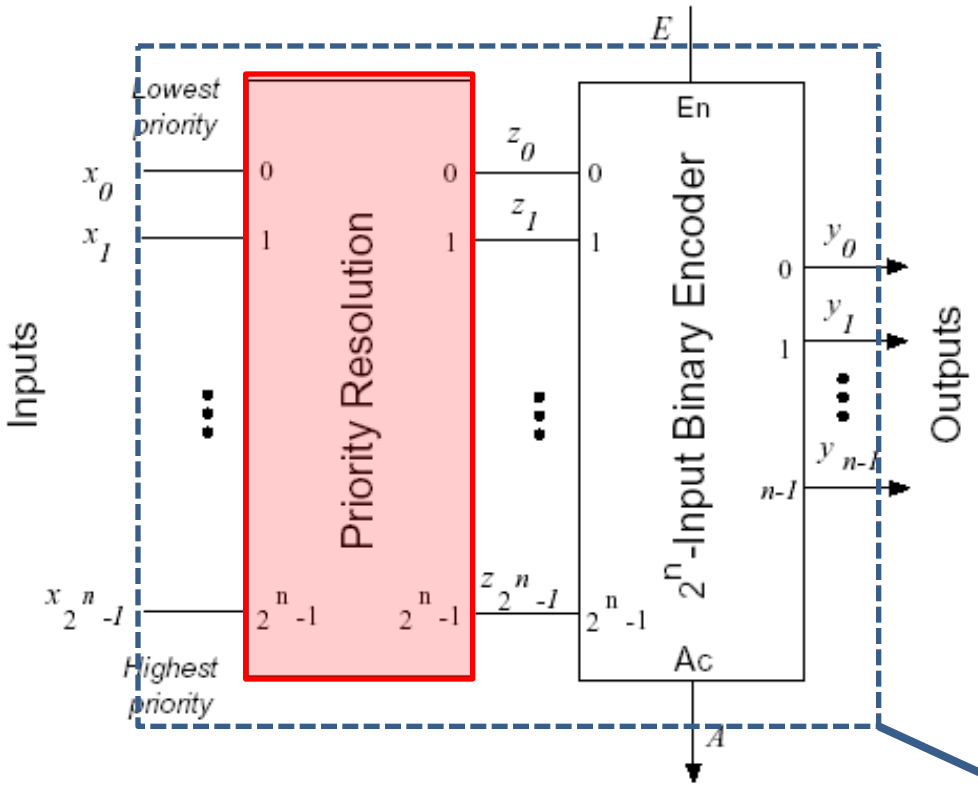


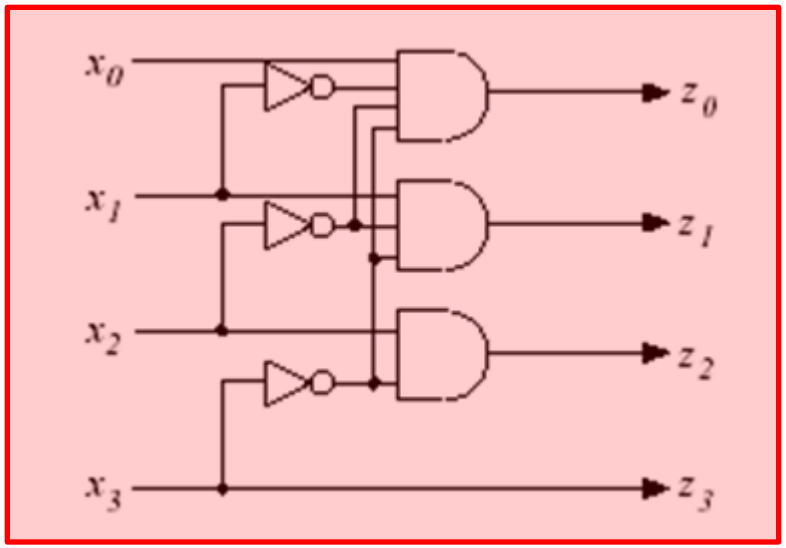
CODIFICADOR (Decimal – BCD)



DECODIFICADOR (BCD – 7 segmentos)

Codificadores de Prioridade

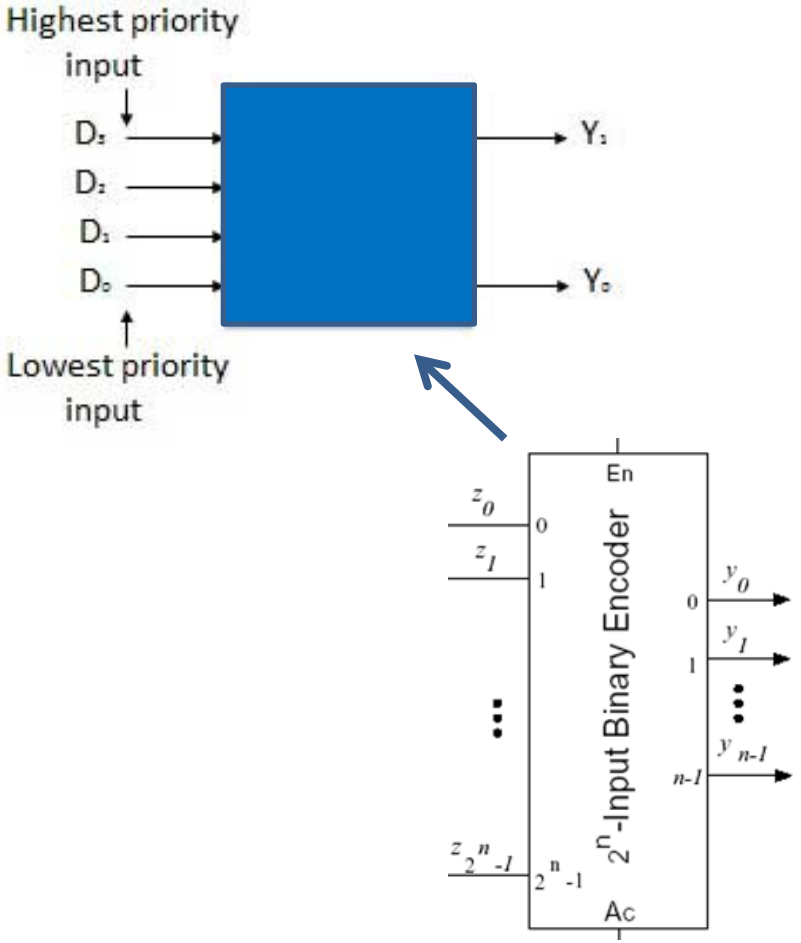




E	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0	y_2	y_1	y_0	A
1	0	0	0	0	0	0	0	1	0	0	0	1
1	0	0	0	0	0	0	1	-	0	0	1	1
1	0	0	0	0	0	1	-	-	0	1	0	1
1	0	0	0	0	1	-	-	-	0	1	1	1
1	0	0	0	1	-	-	-	-	1	0	0	1
1	0	0	1	-	-	-	-	-	1	0	1	1
1	0	1	-	-	-	-	-	-	1	1	0	1
1	1	-	-	-	-	-	-	-	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0	0	0
0	-	-	-	-	-	-	-	-	0	0	0	0

Codificador de Prioridade (4 x 2)

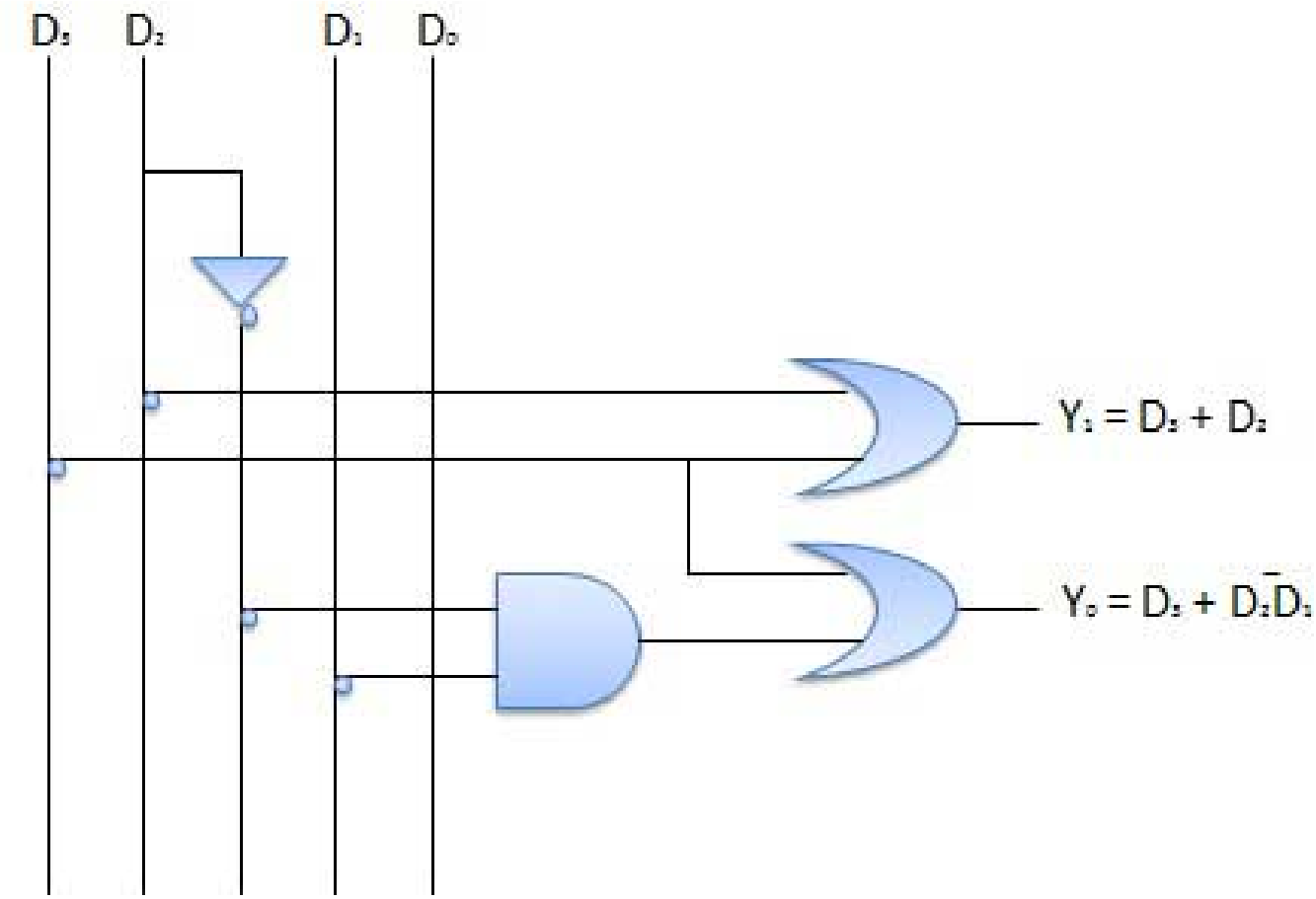
(Priority Encoder)



Highest	Inputs		Lowest	Outputs	
D_3	D_2	D_1	D_0	Y_0	Y_1
0	0	0	0	x	x
0	0	0	1	0	0
0	0	1	x	0	1
0	1	x	x	1	0
1	x	x	x	1	1

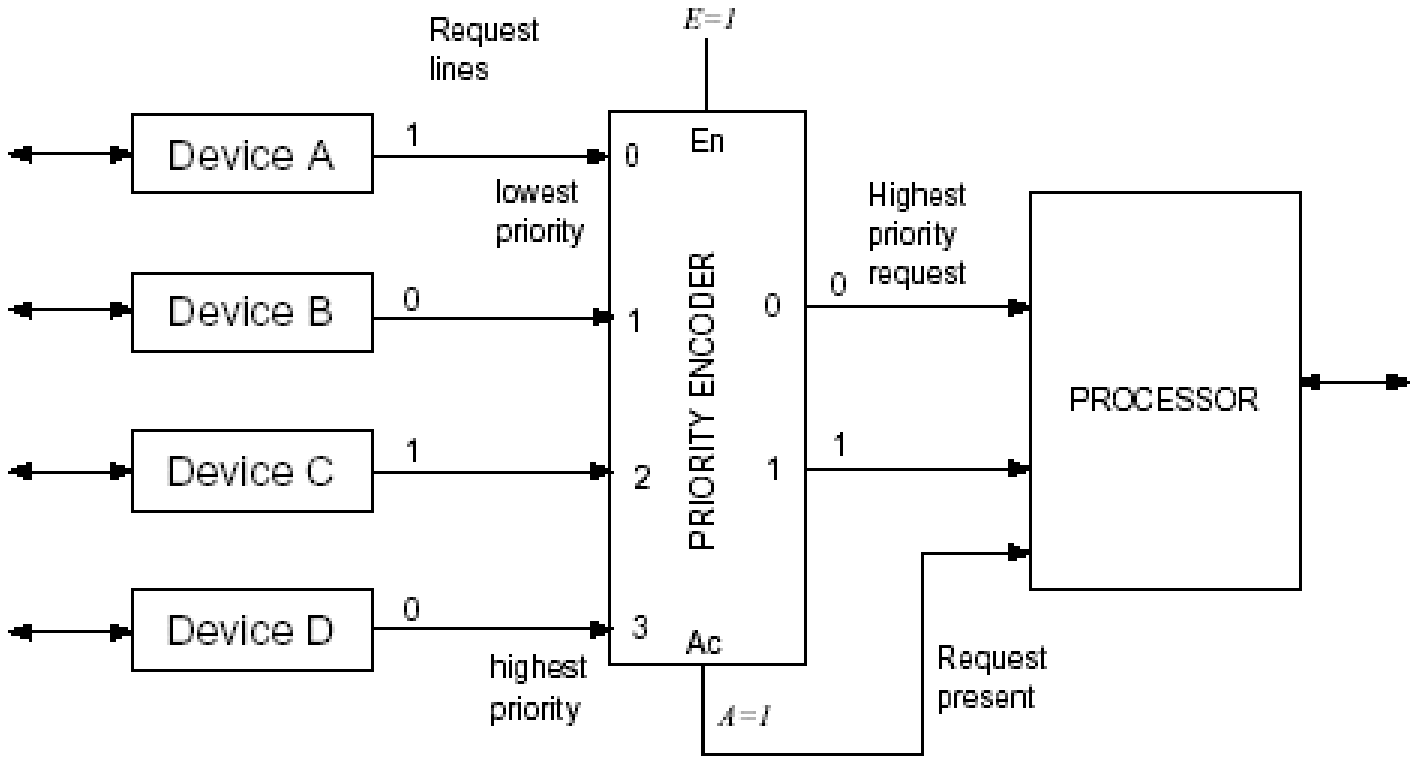
Codificador de Prioridade (4 x 2)

(Priority Encoder)



Codificadores de Prioridade:
Exemplo de Aplicação

- Exemplo de uso: Pedido de Interrupção em um μP (microprocessador).

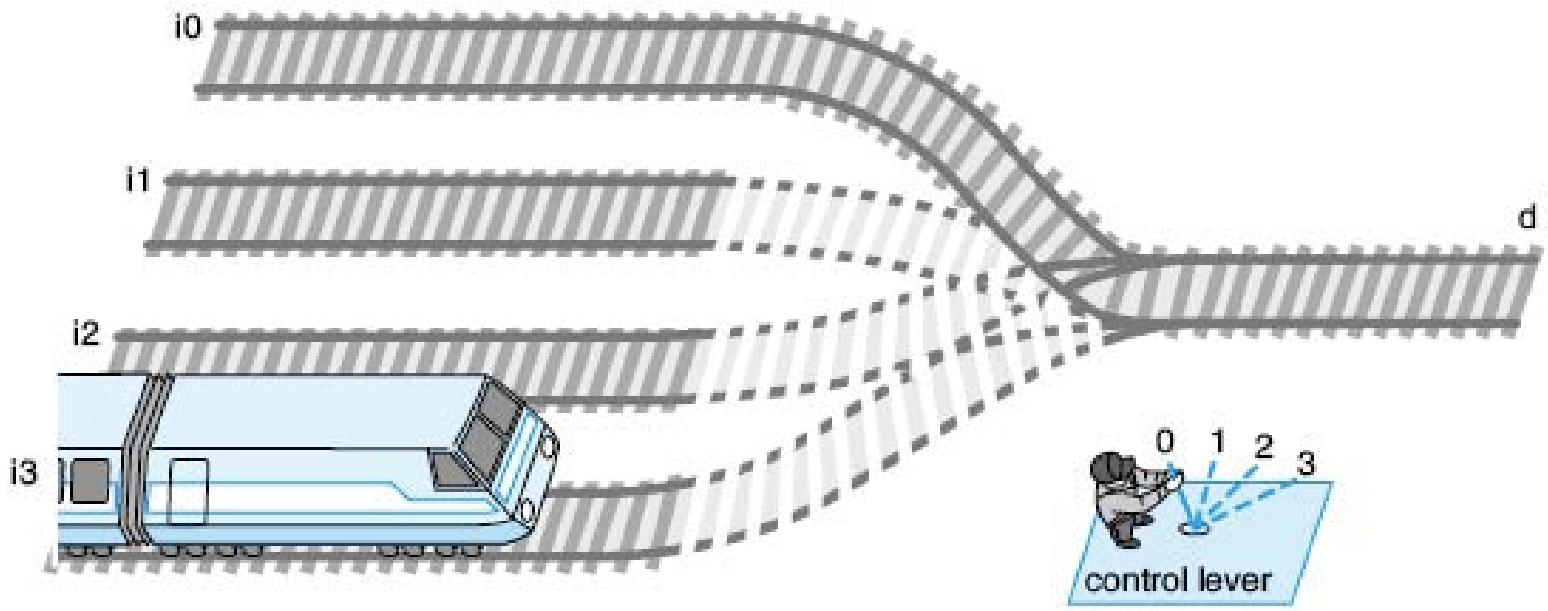


SELETORES:

MULTIPLEXADORES (*MUXes*) / DEMULTIPLEXADORES (DE*MUXes*)

Multiplexadores (MUXes)

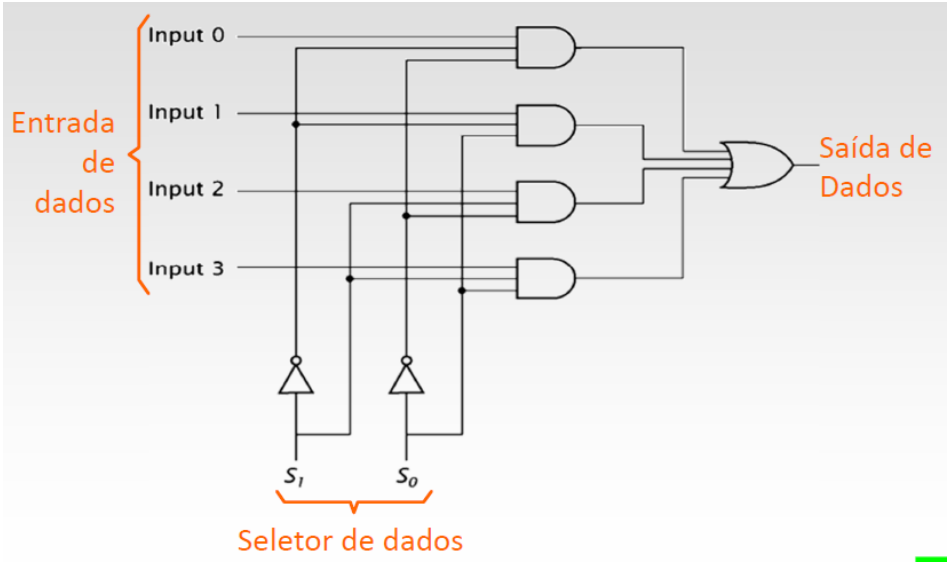
- Um multiplexador Mx1 tem M entradas de dados e 1 saída
- Permite que apenas 1 das entradas seja passada para a saída → **MUX = seletores**

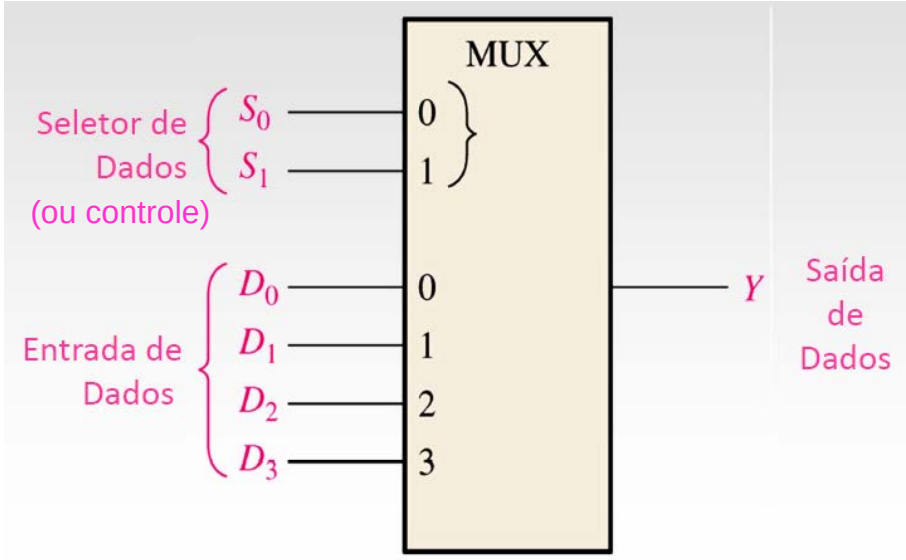


Multiplexadores (MUXes)

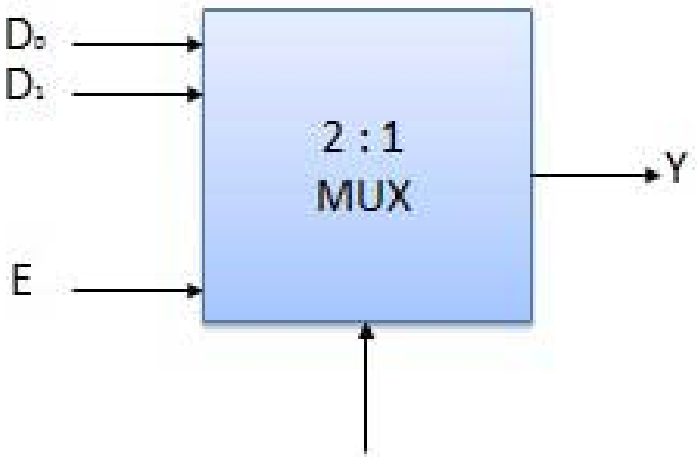
- Um multiplexador (MUX) é circuito lógico que recebe diversos dados digitais de entrada e seleciona um, em determinado instante, para transferi-lo para a saída.
- O envio do dado para a saída é controlado por seletores.
- Para 2^n linhas de entradas, são necessárias n linhas de seleção.

bits (entrada) de controle		saída	
S_1	S_0	Y	
0	0	D_0	dado de entrada
0	1	D_1	
1	0	D_2	
1	1	D_3	





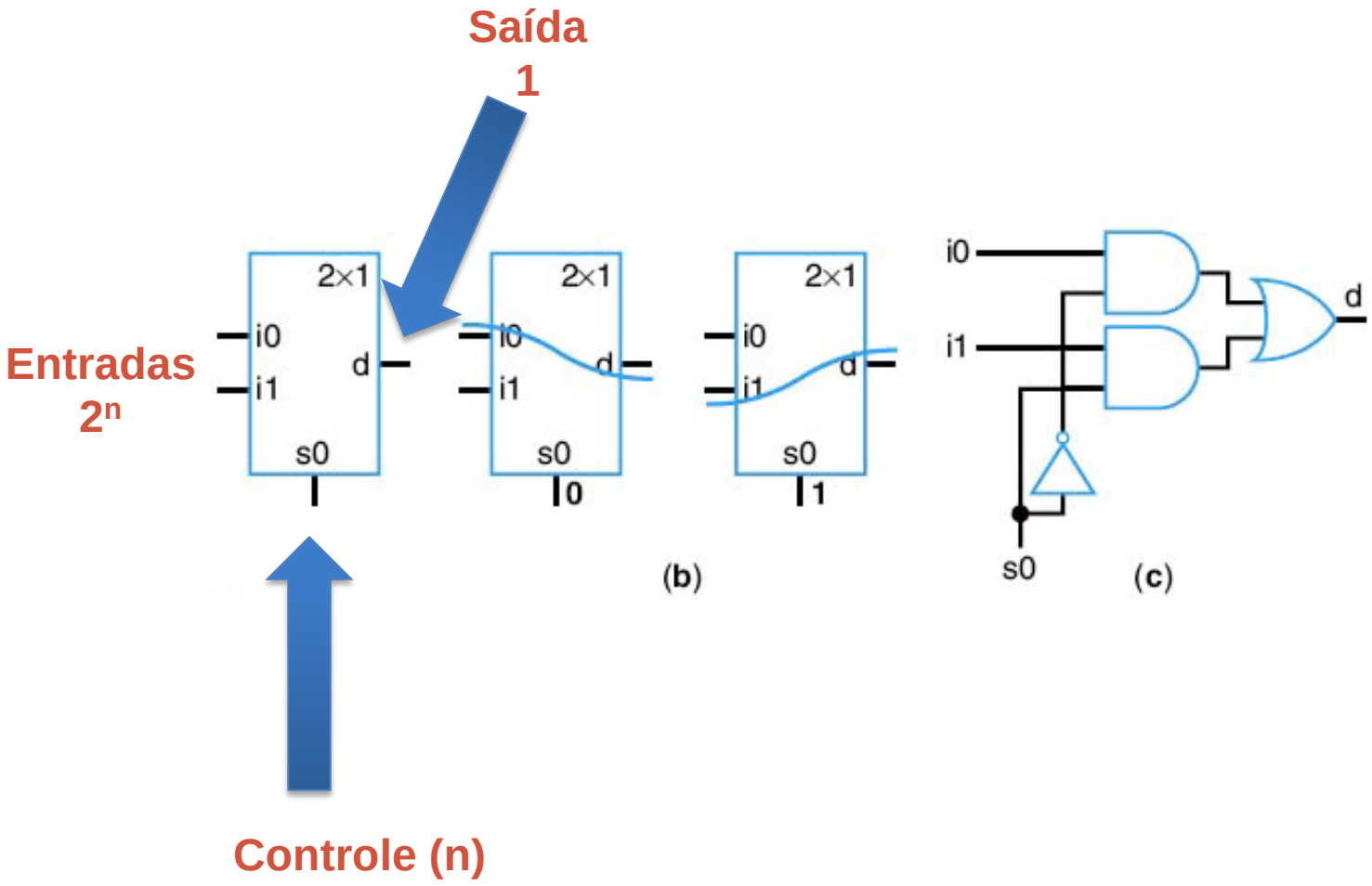
MUX 2x1



Enable	Select	Output
E	S	Y
0	x	0
1	0	D_0
1	1	D_1

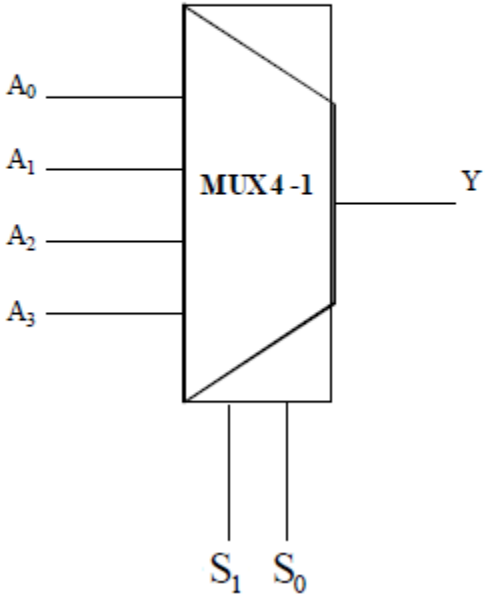
x = Don't care

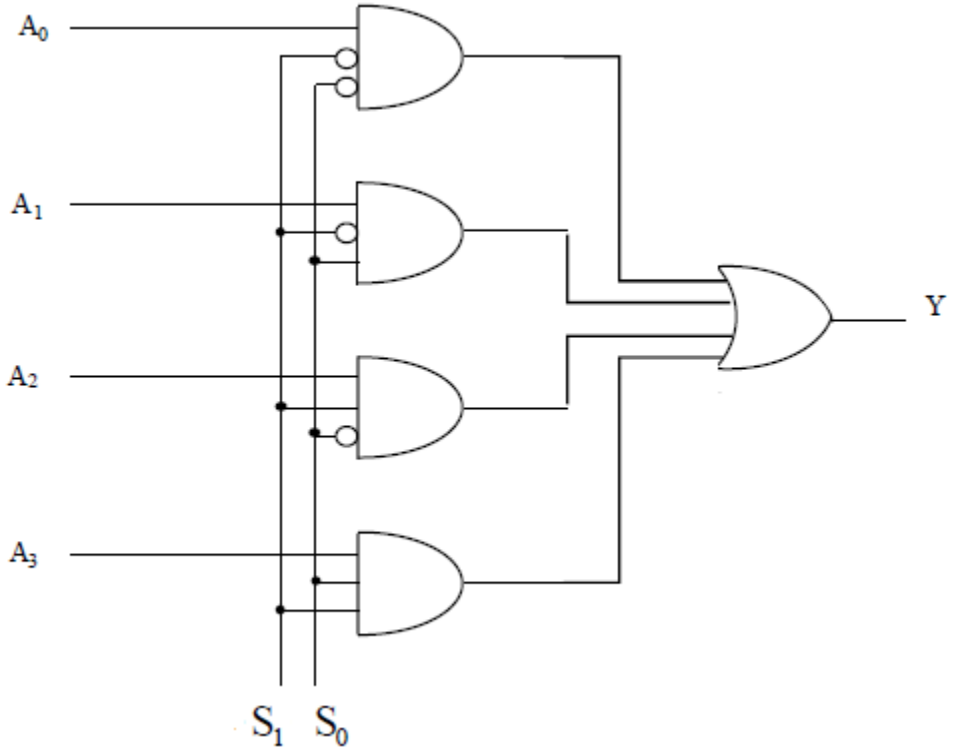
MUX 2x1



MUX 4x1

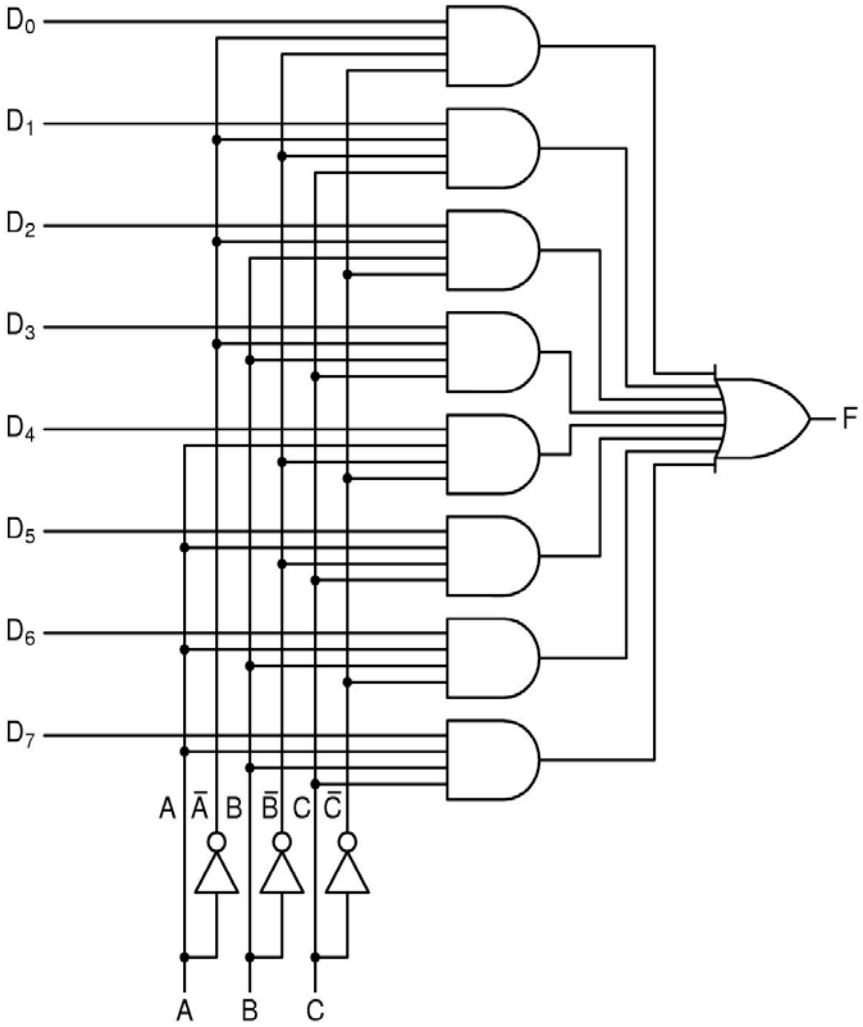
endereço	variáveis de seleção		saída
	S_1	S_0	Y
0	0	0	A_0
1	0	1	A_1
2	1	0	A_2
3	1	1	A_3



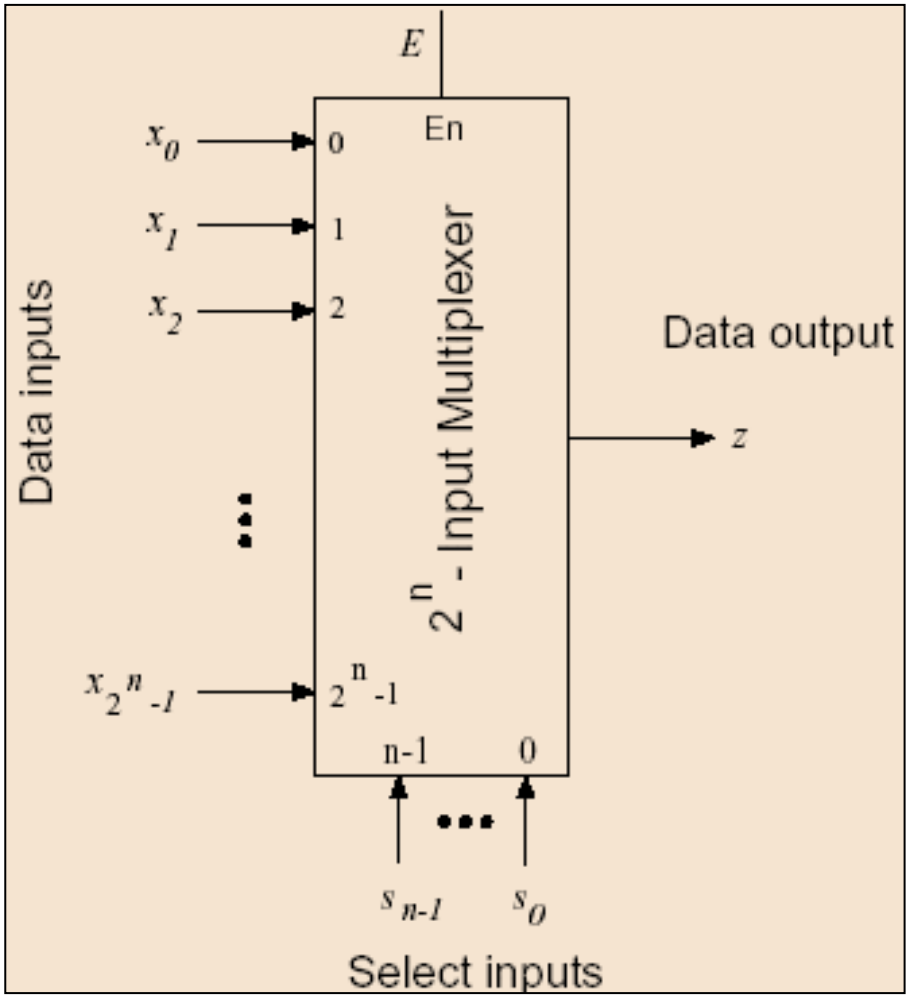


$$Y = \overline{S_0} \cdot \overline{S_1} \cdot A_0 + \overline{S_0} \cdot S_1 \cdot A_1 + S_0 \cdot \overline{S_1} \cdot A_2 + S_0 \cdot S_1 \cdot A_3$$

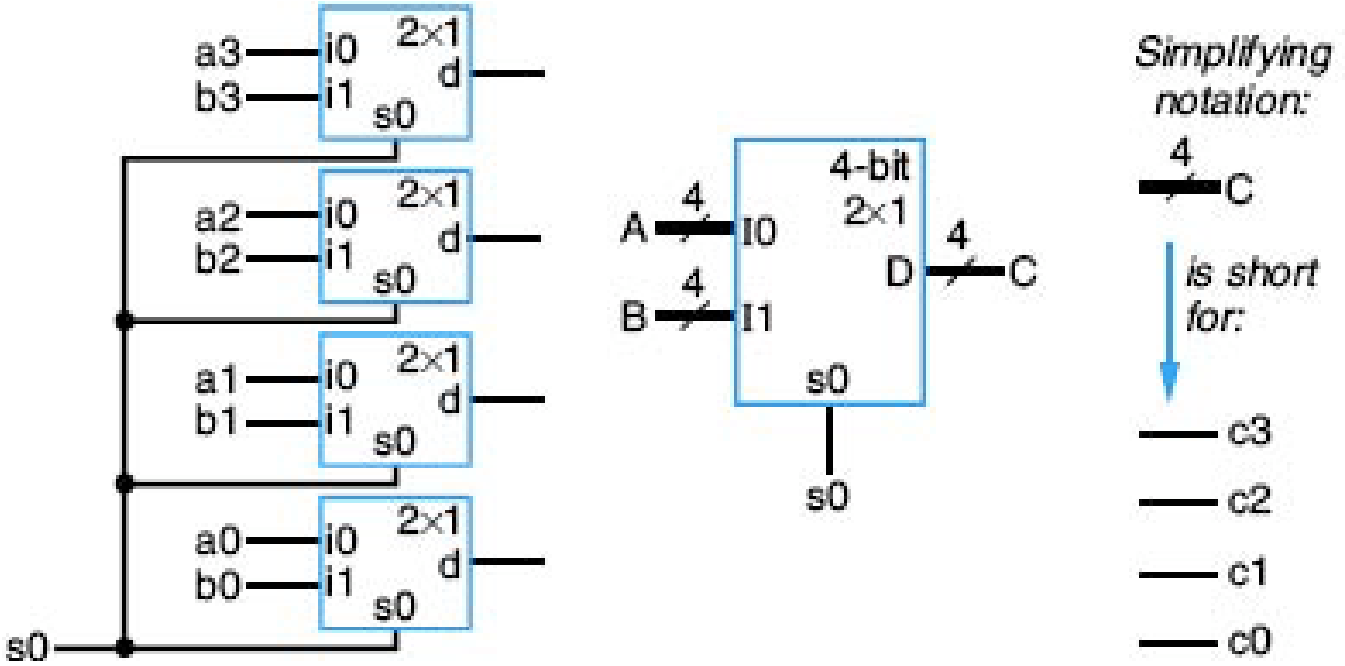
MUX 8x1



Multiplexadores (Generalizando...)



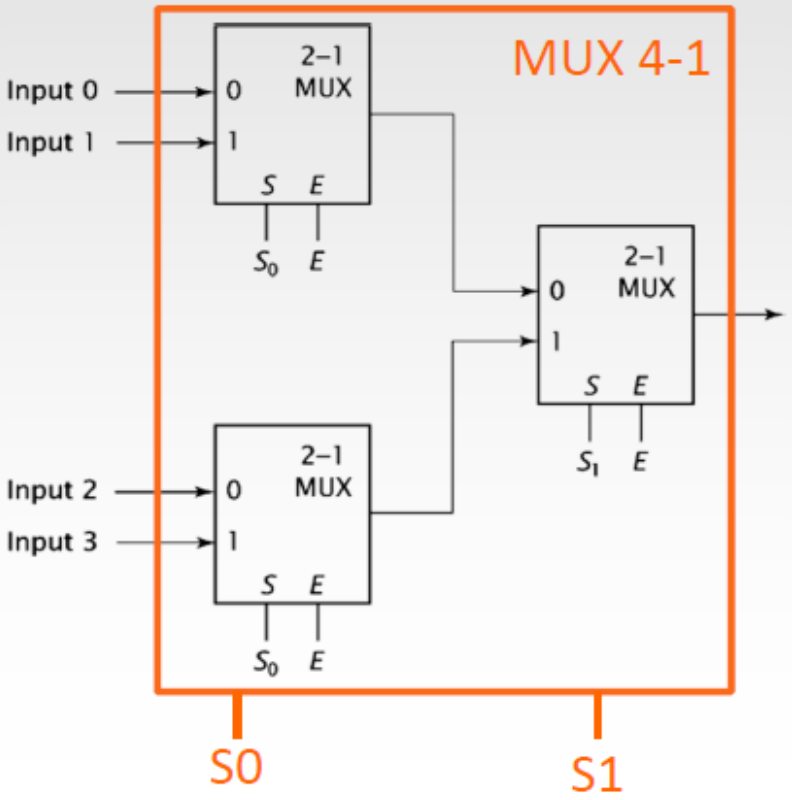
MUX 2x1 de **N bits**



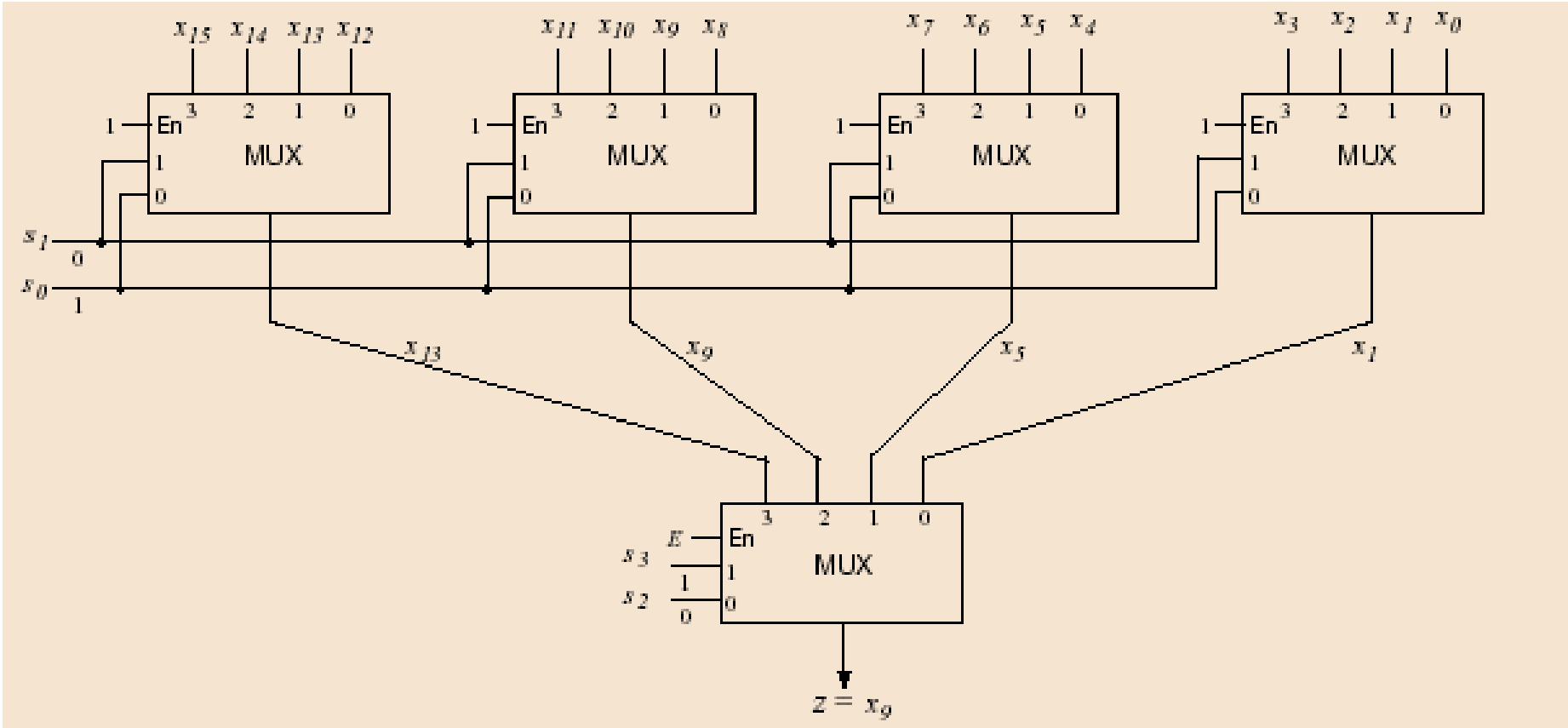
Cascateamento de MUXes

- MUXes com pequeno número de entradas podem ser cascateados para compor um MUX com maior número de entradas:

MUX 4×1 a partir de MUX 2×1



Cascateamento de MUXes

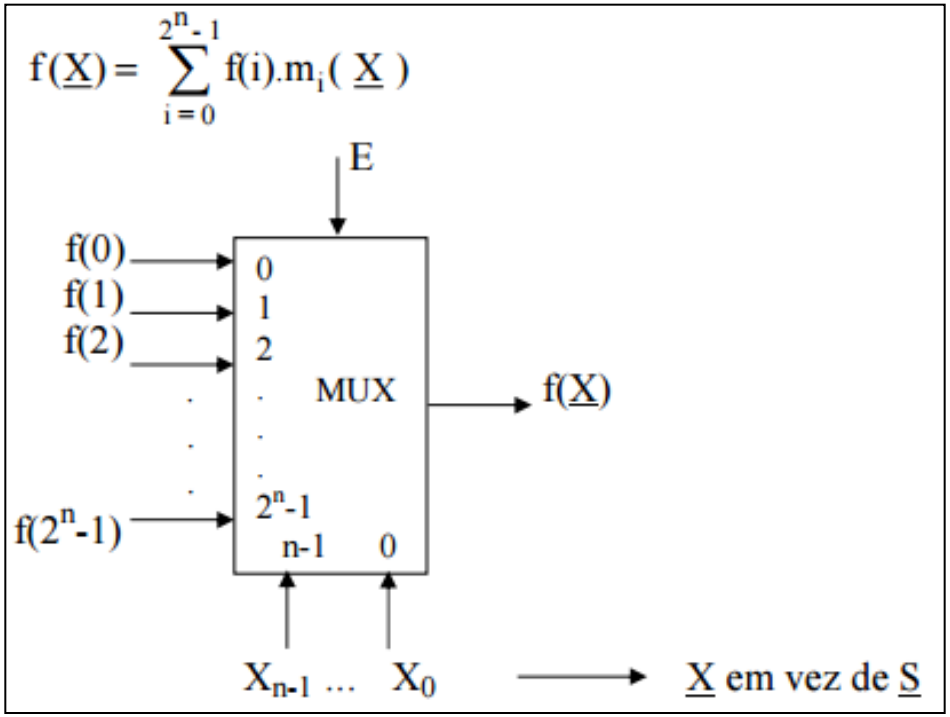


Implementação de Funções com MUX

Um **MUX de 2^n entradas de dados** pode ser usado para implementar qualquer **função de n variáveis**

⇩

MUX - módulo universal



Implementação de Funções com MUX

Um **MUX de 2^n entradas de dados** pode ser usado para implementar qualquer **função de n variáveis**

⇩

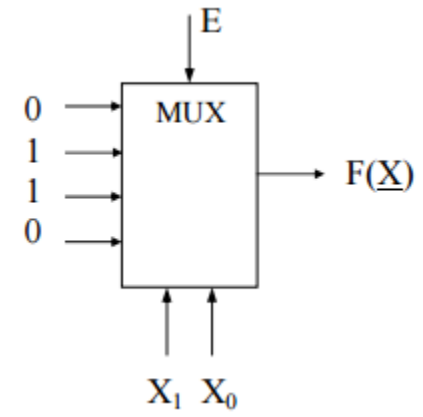
MUX - módulo universal

Exemplo 1:

Tabela de verdade:

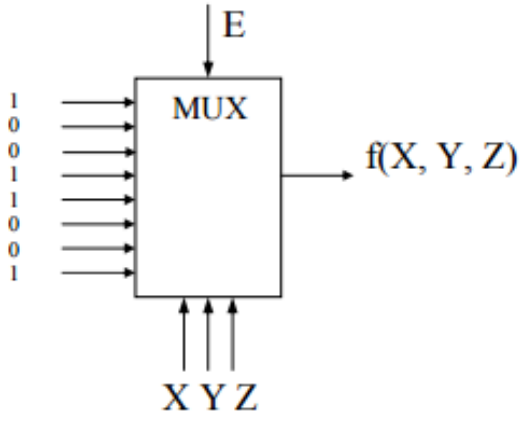
X_1	X_0	f(X)
0	0	0
0	1	1
1	0	1
1	1	0

MUX:



Exemplo 2:

$$f(X,Y,Z) = \sum m(0, 3, 4, 7)$$



Exemplo 3: – Somador completo para 2 bits implementado com 2 MUXs 8x

- Três entradas binárias X, Y e C_{in} (Carry in)
- Duas saídas S (Sum) e C_{out} (Carry out)

Tabela de verdade:

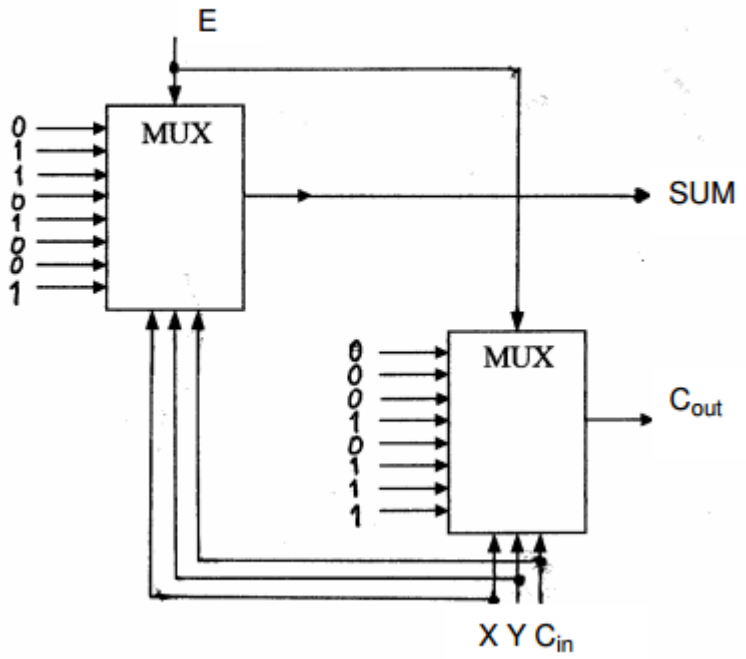
ENTRADAS			SAÍDAS	
X	Y	CARRY IN	SUM	CARRY OUT
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Implementando um Somador com MUXes

$$SUM = \overline{X}.\overline{Y}.C_{in} + \overline{X}.Y.\overline{C_{in}} + X.\overline{Y}.\overline{C_{in}} + X.Y.C_{in}$$

$$C_{out} = \overline{X}.Y.C_{in} + X.\overline{Y}.C_{in} + X.Y.\overline{C_{in}} + X.Y.C_{in}$$

Circuito com MUXs do somador completo:

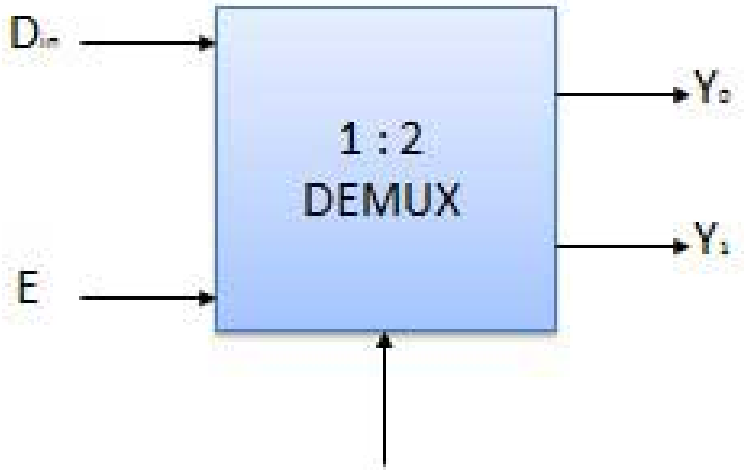


Demultiplexadores (DEMUXes)

- Um multiplexador 1xM tem 1 entrada de dados e M saídas
- Permite que apenas que a saída selecionada receba a entrada

DEMUX 1:2
DEMUX 1:4
DEMUX 1:8
DEMUX 1:16

Demultiplexador (DEMUX)



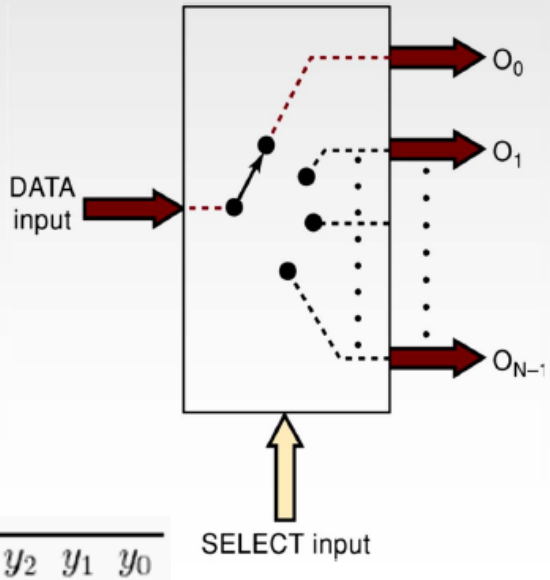
Enable	Select	Output
E	S	Y0 Y1
0	x	0 0
1	0	0 Din
1	1	Din 0

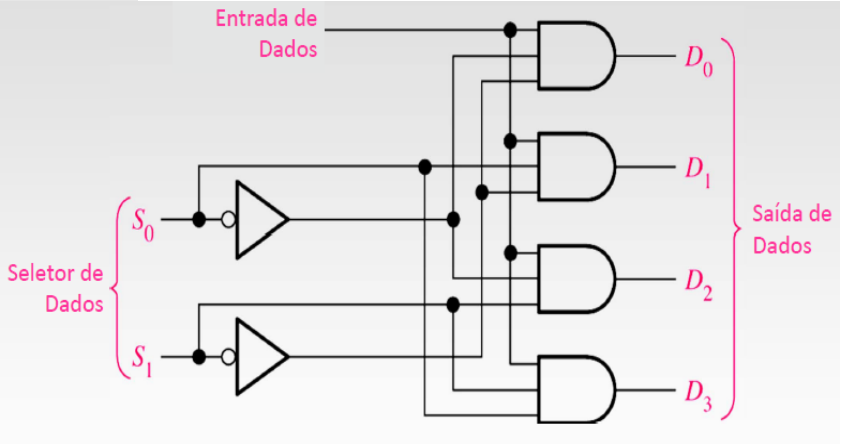
x = Don't care

DEMUX 1:2

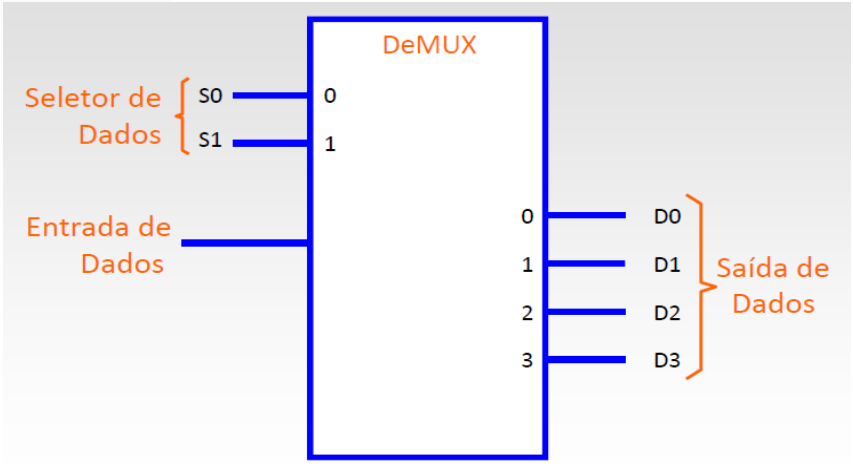
Demultiplexadores (DEMUXes)

- Executa a tarefa inversa do multiplexador.
- Direciona a informação de sua única entrada para uma de suas 2^n saídas, por meio de n variáveis de seleção.

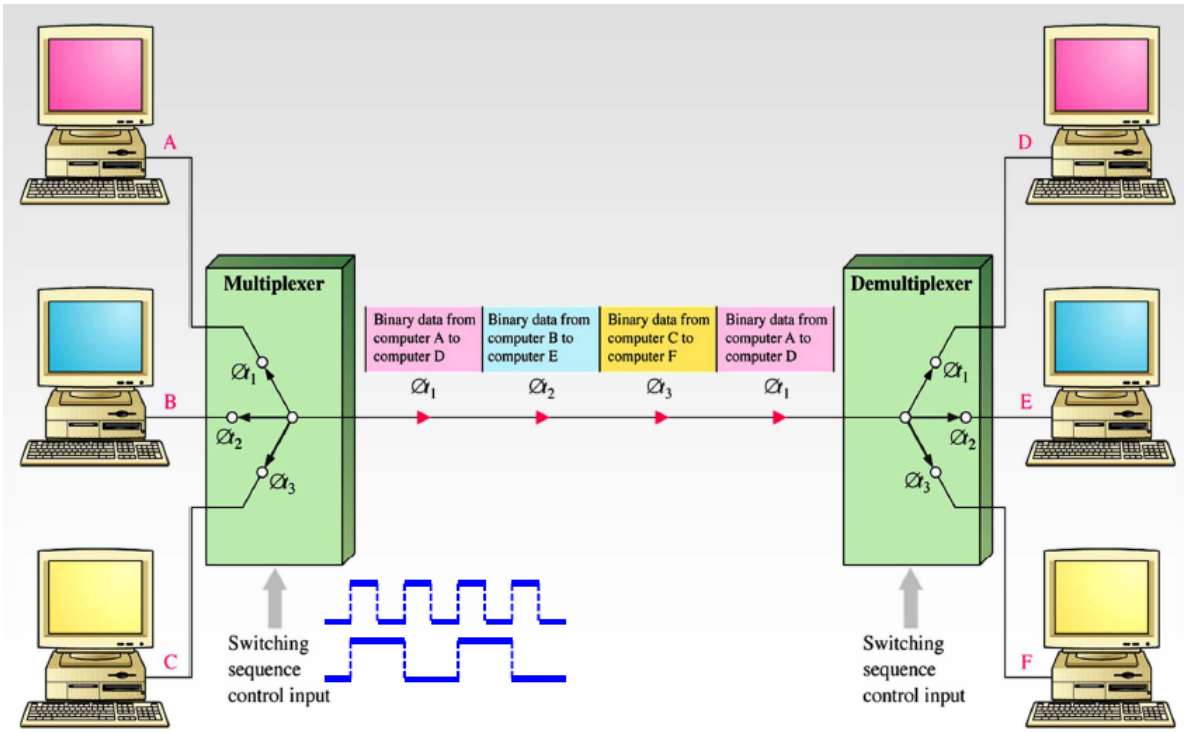


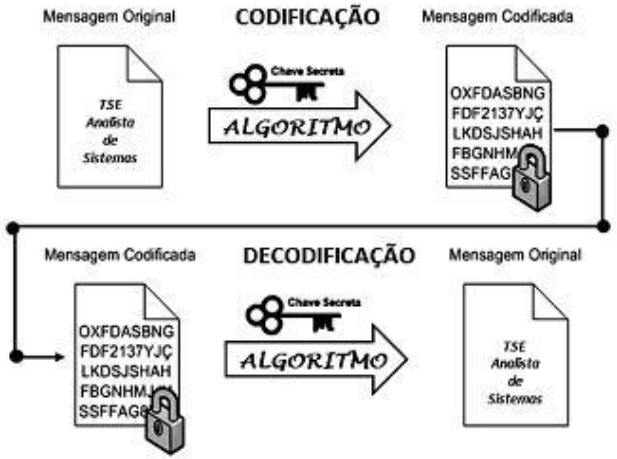


E	s_1	s_0	s	y_3	y_2	y_1	y_0
1	0	0	0	0	0	0	x
1	0	1	1	0	0	x	0
1	1	0	2	0	x	0	0
1	1	1	3	x	0	0	0
0	-	-	-	0	0	0	0



Aplicação (CODE-DECODE / MUX-DEMUX)





ALGUNS EXERCÍCIOS

vide Capítulo 5 do Livro “Elementos da Eletrônica Digital”, IDOETA & CAPUANO

Exercícios

(IDOETA & CAPUANO Cap 5 – Pág. 229 em diante)

1 - Desenhe um sistema somador para 2 números de 2 bits apenas com blocos de Somadores Completos.

5.6.16 - Utilizando blocos de Somadores Completos, elabore um sistema subtrator para 2 números de 2 bits.

5.6.17 - Utilizando blocos de Somadores Completos, elabore um sistema para 2 números de 2 bits que faça soma ou subtração, conforme o nível aplicado a uma entrada de controle M (M = 0 → soma e M = 1 → subtração).

Dúvidas ??

OBRIGADO PELA ATENÇÃO

Prof. Victor M. Miranda